DEFENCE
SCIENCE & TECHNOLOGY

Mar 2003

Velocity Filtering for Target Detection and Track Initiation

Matthew Dragovic

DSTO-TR-1406

DEFENCE
SCIENCE & TECHNOLOGY

Velocity Filtering for Target Detection and Track Initiation

Matthew Dragovic

Weapons Systems Division
Systems Sciences Laboratory

DSTO-TR-1406

ABSTRACT

The velocity filter is a variation of the 3D matched filter. Velocity filtering applies a constraint in the form of assuming that targets will have a constant velocity over the integration period of the filter. Velocity filters are applied over multiple frames of data and are able to detect low Signal-to-Noise Ratio (SNR) targets that would otherwise be undetectable using conventional 'single look' detection techniques. This report derives, discusses and assesses the performance of the velocity filter technique.

RELEASE LIMITATION

Approved for public release

Velocity Filtering for Target Detection and Track Initiation

Executive Summary

Track before detect techniques combine the results from multiple snapshots in time called 'frames' to detect targets in low Signal-to-Noise Ratio (SNR) environments that would otherwise be undetectable using conventional 'single look' techniques. Velocity filtering is a track before detect technique in which an assumption is made that the target motion is of a constant velocity over the integration period of the velocity filter. This report concentrates on the velocity filtering track before detect technique.

The velocity filter is a particular type of multi-dimensional matched filter. The report begins by deriving the matched filter and from here the velocity filter is derived and compared to the matched filter. Simulations were undertaken to test the performance of the velocity filter. The simulations show significant improvements in detection probabilities when the velocity filter technique is employed. This comes at a cost however, as the velocity filter technique requires a significant amount of processing power and memory. Continual increases in processing power and reduced memory costs are seeing track before detect techniques becoming more commonplace.

This study was performed in 2001 as part of a Masters degree in Mathematical Sciences (Signal and Information Processing) that the author undertook in 1999 - 2001 through the Cooperative Research Centre for Sensor Signal and Information Processing (CSSIP) and the University of Adelaide.

Author

Matthew Dragovic
Weapons Systems Division

Matthew Dragovic graduated with a bachelor of Engineering (Electrical & Electronic) with first class honours from the University of Adelaide in 1998. He joined the Defence Science and Technology Organisation in 1998 as a Professional Officer Class 1, working in the Weapon Systems Analysis (WSA) group in Weapon Systems Division (WSD). Whilst with WSA, Matthew's work included simulation and modelling of weapon systems and the development of software to assist in weapon systems analysis. In 2000 Matthew joined the Radio Frequency Seekers (RFS) group in WSD as a Professional Officer Class 2. His work in RFS has been concerned primarily with the performance evaluation of the AMRAAM weapon. In 2001 Matthew completed a Masters of Mathematical Sciences (Signal and Information Processing) degree through the University of Adelaide. Matthew's professional interests include signal processing, radar and electronic warfare. He enjoys tennis, water skiing, surfing and golf.

Contents

1. Introduction

In conventional tracking techniques, tracks are initiated on the basis of a single detection. In low Signal-to-Noise Ratio (SNR) environments these techniques are inadequate and an alternative must be found. Track Before Detect (TBD) techniques attempt to address the track initiation problem.

From the outset it is important to define the terms noise and clutter. The definitions below are taken from [11]. Clutter is defined as unwanted returns from ground, precipitation, or chaff. In fact clutter returns are not always unwanted. For example in SAR imaging, ground returns are the ones of interest. For target detection however, clutter will interfere with the return from the target of interest and hence is always unwanted. Noise is usually random and it consists of electrical or electromagnetic energy that interferes with the detection of wanted signals. Both clutter and noise can interfere with target detection.

Unlike noise, clutter is structured. Pre-processing the data can reduce the amount of correlation in the clutter. The clutter rejection component of the 3D matched filter also helps reduce the correlation of the clutter [3] [4]. Thus, the clutter becomes more 'noise like' in nature and adds incoherently over multiple frames. In this report it is assumed that pre-processing the data and applying the matched filter to the data, will have the effect of removing the correlation of the clutter leaving only a noise like background. It is assumed that the noise at each pixel has a zero mean Gaussian distribution, independent of neighbouring pixels. Hence the spatial correlation of the noise is zero.

TBD techniques involve combining the results from multiple frames. A frame is a snapshot in time of the seeker's field of view. Multi-frame integration occurs over a hypothesised target trajectory. The target energy adds from frame to frame unlike the background noise and clutter. Hence an integration gain occurs and targets which are undetected in single frames become detectable when frames are combined for a given system false alarm rate. TBD techniques exploit the differences in temporal/spatial statistics between clutter and targets.

Velocity filtering is a subset of the 3D matched filter [2]. It is perhaps the simplest TBD technique to understand and implement. Velocity filtering applies a constraint in the form of assuming that targets will have a constant velocity over the integration period of the filter. Since the exact target velocity is unknown a velocity filter bank is used to cover the possible target velocities. An overview of other TBD techniques is given in Reference 5.

3D matched filtering algorithms were originally developed in Reference 2. This paper is the starting point for a discussion of velocity filtering. A further description of the velocity filter is given in Reference 4, which presents a mathematical description of the velocity filter and evaluates its performance via the use of theoretical Receiver Operating Characteristic (ROC) curves.

A major drawback of the velocity filter is that it can require a large amount of memory and considerable computational costs. Reference 3 discusses methods that can be employed to reduce these computational costs and memory, such as sequential velocity filtering and single bit velocity filtering. A technique to reduce the number of computations via projecting 3D space onto two dimensions prior to applying the velocity filter is discussed in Reference 7.

TBD techniques are of particular use in low SNR, high clutter environments such as Anti Ship Missile Defence and periscope detection. With faster, cheaper and smaller processors TBD techniques are likely to become more prolific in the future.

This report concentrates on the velocity filtering TBD technique. The aims of the project were to:

- Develop an understanding of various TBD techniques.
- Develop a strong understanding of the mathematics and implementation issues for the matched and velocity filters.
- Use MATLAB to simulate the velocity filter and compare its performance to theory.

Velocity filtering techniques can be applied to both (Infra Red) IR and (Radio Frequency) RF sensors. This report concentrates on the use of velocity filtering for IR sensors where the input to the filter is a series of images.

In order to understand the mathematics and concepts behind the velocity filter it is a prerequisite to understand both the single and multidimensional matched filter. Chapter 2 derives the 1D and 3D matched filters and assesses their performance. A derivation of the velocity filter with appropriate comparison to the matched filter follows in Chapter 3. Chapter 3 also discusses the performance of the velocity filter and presents two alternative forms that aim to reduce the computational costs and memory requirements.

A one dimensional velocity filter was simulated and its performance is described in Chapter 4. One dimensional velocity filters permit the target to move in a single spatial dimension. The decision was made to simulate a single dimensional velocity filter given the short time frame of the project and the extra complexity involved in simulating a two dimensional filter. The same theory that applies to two dimensional velocity filters is also applicable to the single dimensional case, so a realistic appreciation for velocity filtering can be gained by simulating a one dimensional filter.

Two other types of TBD techniques, namely the Hough Transform and the Dynamic Programming Algorithm are discussed in Appendix A. The MATLAB code used to simulate the velocity filter technique is given in Appendix B.

2. Matched Filters

2.1 Introduction

The velocity filter is a matched filter with the additional assumption of constant target velocity. To understand the velocity filter it is important to first understand the matched filter. This chapter discusses the matched filter and introduces several aspects of the matched filter that are of particular importance to the velocity filter. The chapter begins with the derivation of a single dimensional matched filter and then extends the results to a three dimensional filter. The velocity filter has at least one spatial and time dimension thus multidimensional matched filters are discussed in this section.

The chapter begins by deriving the test statistic of a common detector, the Neyman-Pearson detector [1] [5]. The matched filter is then discussed and it is found that the output of the matched filter is equivalent to the Neyman-Pearson test statistic. The output SNR of the matched filter is then derived and it is found that the larger the SNR the better the probability of detection, as expected.

2.2 The Neyman-Pearson Detector

Consider the case of detecting a known signal (known in both shape and energy) in Gaussian noise. Under Neyman-Pearson criteria [1] the aim is to maximise the probability of detection subject to a constant false alarm probability. The probability of detection is the probability of correctly declaring a target to be present at a particular time and location. The probability of false alarm is the probability of declaring a target present when in fact there is no target. The detector that results from these assumptions is the matched filter. The important point to note for the matched filter to work optimally (that is to produce the maximum possible output) is that the form of the signal that is to be detected must be known.

Let $s[n]$ be a deterministic signal, $w[n]$ denote zero mean white Gaussian noise with variance σ^2 and $v[n]$ be a measurement for $n = 0,1,\ldots,N-1$. Hence, $s[n]$, $w[n]$ and $v[n]$ are all real scalars. Our aim is to determine whether the measurement contains the signal for $n = 0,1,\ldots,N-1$. Thus there are two hypotheses H_0 (noise only) and H_1 (signal plus noise) defined by

$$H_0 : v[n] = w[n],\ n = 0,1,\ldots,N-1 \qquad \text{eq 2-1}$$

$$H_1 : v[n] = s[n] + w[n],\ n = 0,1,\ldots,N-1 \qquad \text{eq 2-2}$$

The Neyman-Pearson detector decides in favour of H_1 if the likelihood ratio, $L(\mathbf{v})$ exceeds a threshold γ

$$L(\mathbf{v}) = \frac{p(\mathbf{v} \mid H_1)}{p(\mathbf{v} \mid H_0)} > \gamma \qquad \text{eq 2-3}$$

where $\mathbf{v}$ is a measurement vector given by

$$\mathbf{v} = \begin{bmatrix} v[0] & v[1] & \ldots & v[N-1] \end{bmatrix}^T$$

As the noise is Gaussian the likelihood functions for hypotheses H_0 and H_1 are

$$p(\mathbf{v} \mid H_1) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left[-\frac{1}{2\sigma^2} \sum_{n=0}^{N-1} (v[n] - s[n])^2 \right] \qquad \text{eq 2-4}$$

$$p(\mathbf{v} \mid H_0) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left[-\frac{1}{2\sigma^2} \sum_{n=0}^{N-1} v^2[n] \right] \qquad \text{eq 2-5}$$

Substituting the above likelihood expressions into Equation 2-3

$$L(\mathbf{v}) = \exp\left[-\frac{1}{2\sigma^2} \left(\sum_{n=0}^{N-1} (v[n] - s[n])^2 - \sum_{n=0}^{N-1} v^2[n] \right) \right] > \gamma \qquad \text{eq 2-6}$$

It is convenient to take the logarithm of the above expression to remove the exponential term. As the logarithm is a monotonically increasing function, the inequality does not change. The condition in Equation 2-6 then becomes

$$-\frac{1}{2\sigma^2} \left(\sum_{n=0}^{N-1} \left(v^2[n] - 2v[n]s[n] + s^2[n] - v^2[n] \right) \right) > \ln \gamma \qquad \text{eq 2-7}$$

Rearranging the above inequality

$$\frac{1}{\sigma^2} \sum_{n=0}^{N-1} v[n]s[n] - \frac{1}{2\sigma^2} \sum_{n}^{N-1} s^2[n] > \ln \gamma \qquad \text{eq 2-8}$$

As the signal is known, the energy term can be incorporated into the threshold.

$$T(\mathbf{v}) = \sum_{n=0}^{N-1} v[n]s[n] > \gamma' \qquad \text{eq 2-9}$$

where the threshold γ' is given by

$$\gamma' = \sigma^2 \ln \gamma + \frac{1}{2} \sum_{n=0}^{N-1} s^2[n] \qquad \text{eq 2-10}$$

and $T(\mathbf{v})$ is the test statistic for hypotheses H_0 and H_1.

The next section defines the matched filter and derives its output. It is found that the output of the matched filter is equivalent to the test statistic for the Neyman-Pearson detector.

2.3 The Matched Filter Derivation

In this section the output of the matched filter is derived and a simple example is considered. Let $v[n]$, the measurement as defined in the previous section, be the input to an FIR with impulse response $h[n]$, where $h[n]$ is nonzero for $n = 0,1,\ldots,N-1$. The output at time n is

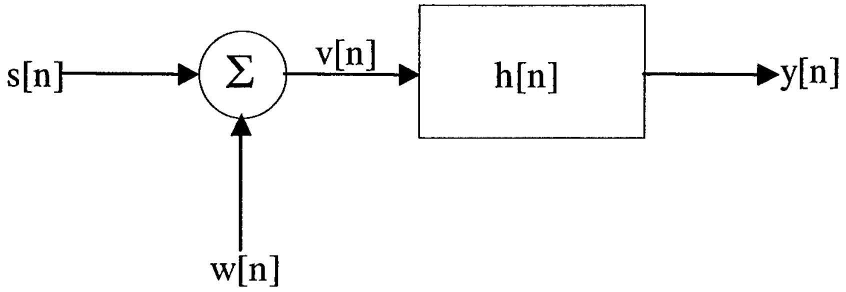


Figure 2-1: Filtering the received signal

$$y[n] = \sum_{k=0}^{n} h[n-k]v[k] \text{, for } n \geq 0 \qquad \text{eq 2-11}$$

The matched filter is so called as it is matched to the signal that it is attempting to detect. The impulse response of the matched filter is simply the 'flipped around' version of the signal [1], i.e.

$$h[n] = s[N-1-n] \qquad \text{eq 2-12}$$

Thus substituting Equation 2-12 into Equation 2-11 yields

$$y[n] = \sum_{k=0}^{N-1} s[N-1-(n-k)]v[k] \qquad \text{eq 2-13}$$

At time $n = N-1$, the above equation becomes

$$y[N-1] = \sum_{k=0}^{N-1} s[k]v[k] \qquad \text{eq 2-14}$$

This is the theoretical maximum output of the matched filter. Note that the above expression is equivalent to the test statistic for the Neyman-Pearson detector (Equation 2-9).

The matched filter impulse response is obtained by flipping the signal s[n] about $n = 0$, and shifting to the right by N-1 samples. An example of the output of a matched filter is shown in Figure 2-2. In this example there are five samples in the signal and hence in the matched filter as well. The maximum output occurs at sample $N-1=4$. This matched filter relies on the fact that the signal begins at $n = 0$. If this is not the case the performance of the detector will be reduced.

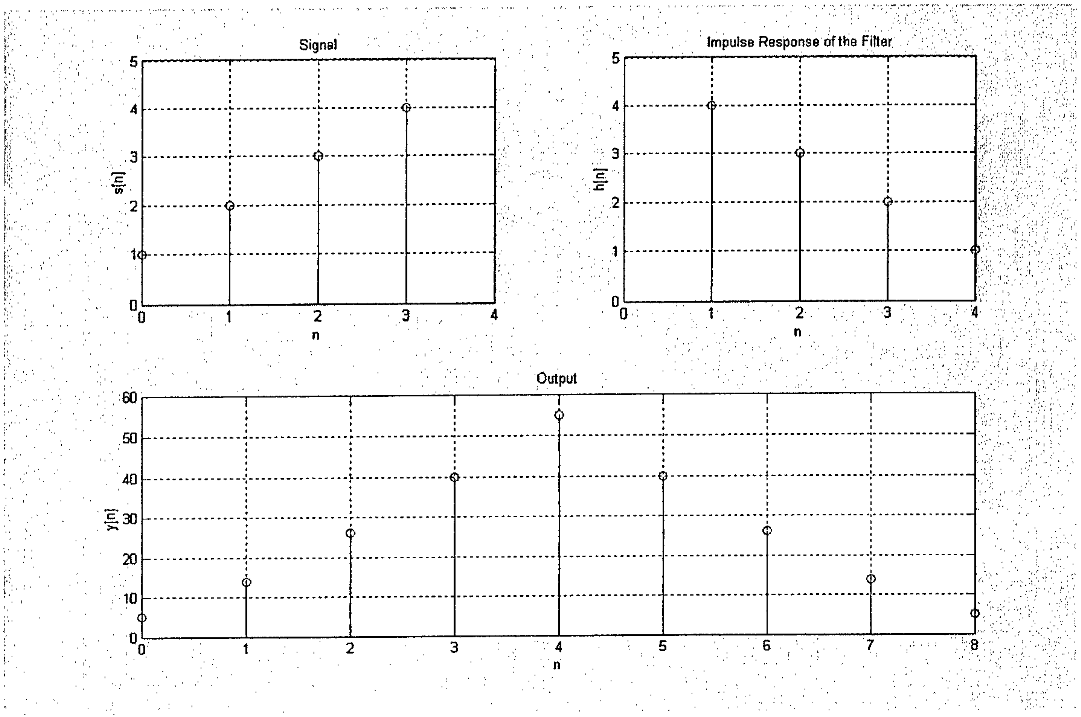


Figure 2-2: Matched filter response for a ramped signal

It is common to express the matched filter in the frequency domain. From Equation 2-12 it follows that the matched filter in the frequency domain is given by

$$H(f) = S^*(f)\exp[-j2\pi f(N-1)] \qquad \text{eq 2-15}$$

where $S^*(f)$ is the complex conjugate of the deterministic signal expressed in the frequency domain.

2.4 SNR Gain of the Matched Filter

The output SNR of the matched filter is an important quantity and is derived in this section. In the following section the importance of the SNR is emphasized when it is found that the distribution of the test statistic is dependent on the SNR.

The output SNR is defined as

$$SNR_{OUT} = \frac{E^2(y[N-1] \mid H_1)}{\mathrm{var}(y[N-1] \mid H_1)} \qquad \text{eq 2-16}$$

where $E^2(\ )$ signifies the use of the expectation operator. Noting that the expectation of white noise is zero and that the signal has constant amplitude and thus zero variance and applying Equation 2-2 and Equation 2-14, leads to the following expression for the output SNR.

$$SNR_{OUT} = \frac{\left(\sum_{k=0}^{N-1} h[N-1-k]s[k]\right)^2}{E\left[\left(\sum_{k=0}^{N-1} h[N-1-k]w[k]\right)^2\right]} \qquad \text{eq 2-17}$$

Defining

$$\mathbf{h} = [h[0] \quad h[1] \quad \ldots \quad h[N-1]]^T$$

$$\mathbf{w} = [w[0] \quad w[1] \quad \ldots \quad w[N-1]]^T$$

yields

$$SNR_{OUT} = \frac{(\mathbf{h}^T\mathbf{s})^2}{E\left[(\mathbf{h}^T\mathbf{w})^2\right]} = \frac{(\mathbf{h}^T\mathbf{s})^2}{\mathbf{h}^T E(\mathbf{w}\mathbf{w}^T)\mathbf{h}} \qquad \text{eq 2-18}$$

$$SNR_{OUT} = \frac{(\mathbf{h}^T\mathbf{s})^2}{\sigma^2\mathbf{h}^T\mathbf{h}} \qquad \text{eq 2-19}$$

where σ^2 is the noise variance. From the Cauchy-Schwarz inequality

$$(\mathbf{h}^T\mathbf{s})^2 \le (\mathbf{h}^T\mathbf{h})(\mathbf{s}^T\mathbf{s}) \qquad \text{eq 2-20}$$

Equality occurs when $\mathbf{h} = c\mathbf{s}$, where c is a constant. This leads to the following expression for the maximum SNR when c is equal to one.

$$SNR_{OUT} = \frac{\mathbf{s}^T\mathbf{s}}{\sigma^2} = \frac{\varepsilon}{\sigma^2} \qquad \text{eq 2-21}$$

where ε is the signal energy. Thus the output SNR of the matched filter is simply the signal energy divided by the variance of the noise.

2.5 Performance of the Matched Filter

In this section a derivation is performed for the Probability Distribution Functions (PDFs) of the matched filter test statistic under each hypothesis H_0 and H_1. An expression is derived for the probability of detection in terms of the probability of false alarm and the output signal to noise ratio of the matched filter and this expression is used to create probability of detection versus SNR plots for various probability of false alarm.

A decision is made in favour of H_1 if the output of the matched filter at $N-1$ is greater than a threshold γ' determined by the probability of false alarm P_{FA}.

$$y[N-1]=\sum_{n=0}^{N-1} v[n]s[n] > \gamma' \qquad \text{eq 2-22}$$

In the previous section it was shown that $y[N-1]$ is equivalent to the test statistic $T(\mathbf{v})$ derived in Section 2.2. In what follows, expressions for the expectation and variance of the test statistic for each hypothesis are derived.

First consider the derivation of the expectation of the test statistic under hypotheses H_0. The noise is white and hence uncorrelated with the signal, thus

$$E(T;H_0)=E\left(\sum_{n=0}^{N-1} w(n)s(n)\right)=0 \qquad \text{eq 2-23}$$

Similarly, under H_1 the expectation is given by

$$\begin{aligned} E(T;H_1) &= E\left(\sum_{n=0}^{N-1}(s(n)+w(n))s(n)\right) \\ &= E\left(\sum_{n=0}^{N-1} s(n)^2\right) \\ &= \varepsilon \end{aligned} \qquad \text{eq 2-24}$$

Thus the two distributions are separated by the signal energy along the test statistic. This is shown in Figure 2-3.

The variance of the test statistic under the noise only hypotheses H_0 is derived below.

$$\begin{aligned} \operatorname{var}(T;H_0) &= \operatorname{var}\left(\sum_{n}^{N-1} w(n)s(n)\right) \\ &= \sum_{n}^{N-1} \operatorname{var}(w(n))s^2(n) \\ &= \sigma^2 \sum_{n=0}^{N-1} s^2(n) \\ &= \sigma^2 \varepsilon \end{aligned} \qquad \text{eq 2-25}$$

Similarly, it can be shown that

$$\operatorname{var}(T;H_1)=\sigma^2\varepsilon \qquad \text{eq 2-26}$$

The test statistic is Gaussian distributed under each hypothesis, with equal variance and different means

$$T \sim \begin{cases} N(0,\sigma^2\varepsilon) \mid H_0 \\ N(\varepsilon,\sigma^2\varepsilon) \mid H_1 \end{cases} \qquad \text{eq 2-27}$$

The above results are shown diagrammatically in Figure 2-3. This figure shows two coloured sections. The blue section illustrates the values of the test statistic for which a false alarm will occur if the input consists of noise only. Conversely the red section illustrates the values of the test statistic for which an actual signal will be missed. The threshold is chosen to be at the point where the two PDFs intersect; this is referred to as Maximum Likelihood detection.

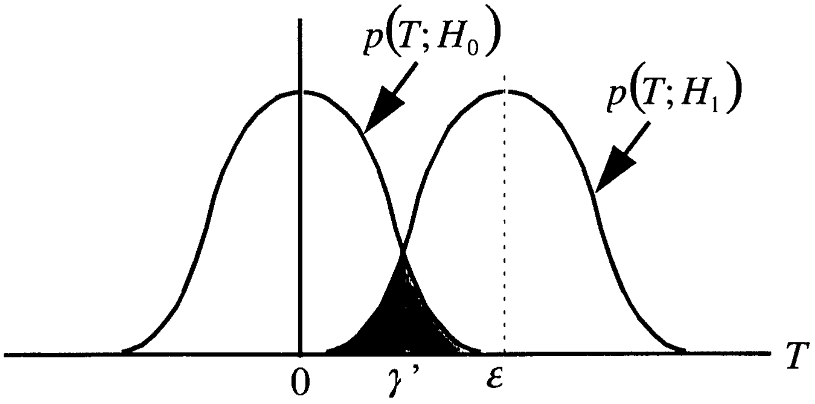


Figure 2-3: PDFs of the matched filter test statistic

Detection performance can be improved in two ways. Firstly increasing the signal energy will increase the separation of the distribution for each hypothesis. However this will increase the variance of each distribution, $\sigma^2\varepsilon$, thus the noise power must be reduced as well to compensate for this. Therefore improved detection performance is obtained by increasing the SNR. This is further emphasized if the test statistic is scaled

$$T' = \frac{T}{\sqrt{\sigma^2\varepsilon}} \qquad \text{eq 2-28}$$

The distribution for the scaled test statistic for each hypothesis is

$$T' \sim \begin{cases} N(0,1) \mid H_0 \\ N(\sqrt{\varepsilon/\sigma^2},1) \mid H_1 \end{cases} \qquad \text{eq 2-29}$$

i.e.

$$T^{'} \sim \begin{cases} N(0,1)\,|\,H_0 \\ N(\sqrt{SNR_{OUT}},1)\,|\,H_1 \end{cases} \qquad \text{eq 2-30}$$

Expressions for the probability of false alarm and the probability of detection given a particular threshold are derived as follows. The probability of false alarm is the probability that the test statistic exceeds the threshold when there is only noise present in the measurement.

$$\begin{aligned} P_{FA} &= \Pr\{T > \gamma' \,|\, H_0\} \\ &= Q\left(\frac{\gamma'}{\sqrt{\sigma^2\varepsilon}}\right) \end{aligned} \qquad \text{eq 2-31}$$

where

$$Q(x) = \int_x^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{t^2}{2}\right) dt = 1 - \Phi(x) \qquad \text{eq 2-32}$$

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp\left(-\frac{u^2}{2}\right) du \qquad \text{eq 2-33}$$

$\Phi(x)$ is the normal cumulative distribution function. It is a monotonically increasing function and hence has an inverse, as does $Q(x)$.

Conversely, the probability of detection is the probability that, in the case of the measurement containing the signal as well as noise, the test statistic exceeds the threshold and a decision is made that a target is present.

$$\begin{aligned} P_D &= \Pr\{T > \gamma' \,|\, H_1\} \\ &= Q\left(\frac{\gamma' - \varepsilon}{\sqrt{\sigma^2\varepsilon}}\right) \end{aligned} \qquad \text{eq 2-34}$$

By rearranging Equation 2-31 a formula can be obtained for the threshold

$$\gamma' = \sqrt{\sigma^2\varepsilon}\,Q^{-1}(P_{FA}) \qquad \text{eq 2-35}$$

Substituting Equation 2-35 into Equation 2-34 gives

$$
\begin{aligned}
P_D &= Q\left(\frac{\sqrt{\sigma^2\varepsilon}Q^{-1}(P_{FA})-\varepsilon}{\sqrt{\sigma^2\varepsilon}}\right)\\
&= Q\left(Q^{-1}(P_{FA})-\sqrt{\frac{\varepsilon}{\sigma^2}}\right)\\
&= Q\left(Q^{-1}(P_{FA})-\sqrt{SNR_{OUT}}\right) \qquad \text{eq 2-36}
\end{aligned}
$$

Thus the probability of detection can be written as a function of the probability of false alarm and the output SNR of the matched filter. Figure 2-4 shows how the P_D changes for different SNR's and P_{FA}. The figure shows that for a given probability of false alarm, the probability of detection increases as the SNR increases. The figure also shows that for a given SNR, the probability of detection cannot be increased without an increase in the probability of false alarm. Consider an example for a Constant False Alarm Rate (CFAR) of 10^{-4}. Table 2-1 shows a comparison of different SNR's required for different probabilities of detection.

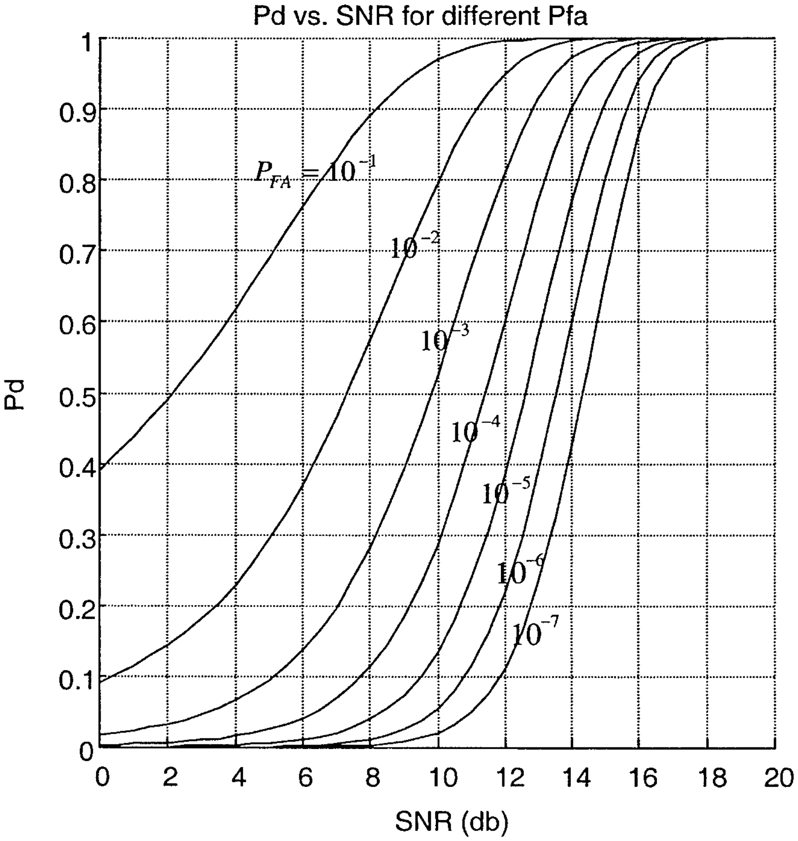


Figure 2-4: SNR required for a specified probability of detection for different probabilities of false alarm.

Table 2-1:SNR required for a specified probability of detection with a false alarm rate of one in 10000.

Specified Probability of Detection	Required SNR (db)
0.8	13.2
0.9	14.0
0.95	14.6
0.99	15.7
0.999	16.7

This section has shown the importance of the SNR on the performance of the matched filter. It is important to note that it is the signal energy and not the shape of the signal that affects detection performance.

2.6 Processing Gain of the Matched Filter

In the velocity filter technique, an improvement in the SNR is obtained by integrating the results from multiple frames. In this section the improvement in the SNR obtained for the matched filter by processing N samples instead of a single sample is investigated. The results obtained here can be compared directly to the results obtained for the velocity filter in Section 3.3.

Consider the case where a constant amplitude signal of size A is present. Using a single sample, the performance is a function of

$$SNR_{IN} = \frac{\varepsilon}{\sigma^2} = \frac{A^2}{\sigma^2} \qquad \text{eq 2-37}$$

If instead the matched filter processes N samples

$$SNR_{OUT} = E\left\{ \frac{\left(\sum_{n=1}^{N} s(n)\right)^2}{\left(\sum_{n=1}^{N} w(n)\right)^2} \right\} = \frac{E\left\{\left(\sum_{n=1}^{N} s(n)\right)^2\right\}}{E\left\{\left(\sum_{n=1}^{N} w(n)\right)^2\right\}} \qquad \text{eq 2-38}$$

as the numerator and denominator are independent.

$$SNR_{OUT} = \frac{E\left\{\left(\sum_{n=1}^{N} A\right)^2\right\}}{E\left\{\left(w(1) + w(2) + \ldots + w(N)\right)^2\right\}} \qquad \text{eq 2-39}$$

The noise samples are uncorrelated with each other, thus

$$SNR_{OUT} = \frac{(NA)^2}{E\{w(1)^2\} + E\{w(2)^2\} + \ldots + E\{w(N)^2\}} \qquad \text{eq 2-40}$$

Now $E\{w(i)^2\} = \sigma^2$ for all i, therefore

$$SNR_{OUT} = \frac{(NA)^2}{N\sigma^2} = N\frac{A^2}{\sigma^2} = N \times SNR_{IN} \qquad \text{eq 2-41}$$

Thus the processing gain obtained by processing N samples is

$$PG = 10\log_{10}\frac{SNR_{OUT}}{SNR_{IN}} = 10\log_{10} N \text{ dB} \qquad \text{eq 2-42}$$

The above has shown that an improvement in detection performance can be obtained by processing multiple samples. This is useful if a probability of detection is required, which is not satisfied given a specific constant false alarm rate. However, using multiple samples may create latency in the detection process, as it is necessary to wait for a period of time to receive then process the samples.

2.7 Three Dimensional Matched Filters

The discussions thus far have dealt with the single dimensional matched filter. The velocity filter is multidimensional. It will have a temporal dimension and at least one spatial dimension. In this section the one dimensional matched filter is extended to three dimensions. The results are used in the derivation of the velocity filter in Chapter 3.

The 3D matched filter [2] is an extension of the 1D matched filter discussed above. It is common to associate the 3D matched filter with measurements of a spatial area over time. The optimal 3D matched filter maximises the SNR for a given target trajectory over a given field of view for a set period of time.

An example of where a 3D matched filter may be used is in the case of an IR seeker, which records images over time and uses these images to detect targets. Assumptions can be made regarding the motion of the target, for example the user may wish to consider targets of constant velocity. The filter that results from this assumption is the velocity filter.

Table 2-2 shows a comparison between the one and three dimensional matched filters. Unlike the 1D filter where the index into the measurement, signal and noise vector is a scalar, the index for the 3D matched filter is a vector of spatial and time components, $\mathbf{r}$.

Table 2-2: A comparison between the 1D and 3D matched filters

	1D Matched Filter	3D Matched Filter
Measurement	$v(n)$	$v(\mathbf{r})$
Signal	$s(n)$	$s(\mathbf{r})$
Noise	$w(n)$	$w(\mathbf{r})$
Noise variance	σ^2	Σ, Power Spectral Density of the background noise
Hypothesis, H_0	$v(n) = w(n)$	$v(\mathbf{r}) = w(\mathbf{r})$
Hypothesis, H_1	$v(n) = s(n) + w(n)$	$v(\mathbf{r}) = s(\mathbf{r}) + w(\mathbf{r})$
Index	$n = 0,\dots,N-1$	$\mathbf{r} = (\mathbf{n}_\mathrm{x}, \mathbf{n}_\mathrm{y}, \mathbf{n}_\mathrm{t})$ $\mathbf{n}_\mathrm{x} = [n_x[0] \quad n_x[1] \quad \dots \quad n_x[N_x - 1]]^T$ $\mathbf{n}_\mathrm{y} = [n_y[0] \quad n_y[1] \quad \dots \quad n_y[N_y - 1]]^T$ $\mathbf{n}_\mathrm{t} = [n_t[0] \quad n_t[1] \quad \dots \quad n_t[N_t - 1]]^T$
Filter Impulse Response	$h(n) = s(N-1-n)$	$h(\mathbf{r}) = s(\mathbf{N} - \mathbf{1} - \mathbf{r})$ $\mathbf{N} = (N_x, N_y, N_t)$ $\mathbf{1} = (1,1,1)$ $h(\mathbf{n}_\mathrm{x}, \mathbf{n}_\mathrm{y}, \mathbf{n}_\mathrm{t}) = s(N_x - 1 - \mathbf{n}_\mathrm{x}, N_y - 1 - \mathbf{n}_\mathrm{y}, N_t - 1 - \mathbf{n}_\mathrm{t})$
Filter Transfer Function	$H(f) = S^*(f) e^{(-j2\pi f(N-1))}$	$H(f) = \dfrac{S^*(f)}{\Sigma(f)} e^{(-j2\pi f r_0)}$
Output	$y(n) = \sum_{k=0}^{n} h(n-k) v(k)$	$y(\mathbf{r}) = \sum_{\Gamma_3} h(\mathbf{r} - \mathbf{u}) v(\mathbf{u})$ $y(n_x, n_y, n_t) = \sum_{u_x=0}^{n_x} \sum_{u_y=0}^{n_y} \sum_{u_t=0}^{n_t} h(n_x - u_x, n_y - u_y, n_t - u_t) v(u_x, u_y, u_t)$
SNR	$SNR_{OUT} = \dfrac{E\ (y[n-1] \mid H_1)}{\mathrm{var}(y[N-1] \mid H_1)}$ $SNR_{OUT} = \dfrac{s^T s}{\sigma^2}$	$SNR_{OUT} = \dfrac{\lvert mean(y(\mathbf{r}_0)) \rvert^2}{\mathrm{var}(y(\mathbf{r}_0))}$ $SNR_{OUT} = \sum_{\Gamma_3} \dfrac{\lvert S(f) \rvert^2}{\Sigma(f)}$ $\mathbf{r}_0$: target location at time t = 0 Γ_3: represents that the sum is performed over 3 dimensions

3. Velocity Filters

3.1 Introduction

The velocity filter is a variation of the 3D matched filter. It assumes a constant target velocity and that the signature of the target remains constant over time. The exact target velocity is unknown and therefore a bank of velocity filters is employed which cover the hypothesized target trajectories. As was shown in the 1D matched filter case, using multiple frames can improve the detection performance of the velocity filter. In fact the concept of a velocity filter relies on the use of multiple frames.

The assumption of a constant target velocity and hence straight-line motion effects the type of targets, the number of frames, the update rate required for each frame and the range of target velocities that the velocity filters must cover. For example in the case of anti-ship missile defence, incoming missiles will be travelling at high speeds. As a result frames will have to be processed in the order of once every second. In the case of periscope detection, submarines travel at a far slower speed and as such, processing of the frames can be delayed to be in the order of several seconds.

As stated above, the velocity filter is a 3D matched filter with some extra requirements concerning the type of targets it is used to detect. The clutter rejection component of the 3D matched filter reduces the correlation of the clutter [3] [4]. Thus the clutter becomes more 'white' in nature and adds incoherently over multiple frames. Over N_F frames the clutter power will increase by a factor of N_F. The target however will be present at the same location in all frames, when the velocity filter is matched to the target velocity, and hence the target power will increase by a factor of N_F^2 over N_F frames. Thus the signal to noise ratio is increased by a factor equal to the number of frames. A mathematical argument is provided to support the above in Section 3.3.

Velocity filters provide the sensor with position and velocity information, which is needed for track initiation. They are useful for detecting multiple targets, as the number of computations is independent of the targets present in the field of view. However, data association logic must be employed in order to associate detections with the correct target. Such logic is not employed in this report, however simulations involving multiple detections were performed and are discussed in Section 4.1.2.

3.2 The Velocity Filter: A Mathematical Description

We begin by forming a mathematical description of the velocity filter. The 3D matched filter is defined in Reference 2 to have the following transfer function:

$$H(f) = \frac{S^*(f)}{\Sigma(f)} \exp(-j2\pi f r_0) \qquad \text{eq 3-1}$$

It is convenient to convert this to the z-domain,

$$H(z) = \frac{S^*(z)}{\Sigma(z)} \qquad \text{eq 3-2}$$

Expanding Equation 3-2 into its spatial and temporal components yields,

$$H(z_x, z_y, z_t) = \frac{S(z_x^{-1}, z_y^{-1}, z_t^{-1})}{\Sigma(z_x, z_y, z_t)} \qquad \text{eq 3-3}$$

where $S(z_x, z_y, z_t)$ is the z-transform of the spatio-temporal signal $s(n_x, n_y, n_t)$ and $\Sigma(z_x, z_y, z_t)$ is the power spectral density of the background clutter expressed in the z-domain. Note that the signal is sometimes called the target function [4]. The pre-processing of the frames that occurs prior to applying the velocity filter has the effect of whitening the background clutter. Thus in all simulations performed in this study the background noise was assumed to be white.

Assuming the target moves with a constant velocity the signal can be redefined to be

$$s(n_x, n_y, n_t) = s(n_x - v_x n_t, n_y - v_y n_t) \qquad \text{eq 3-4}$$

where v_x and v_y are the components of constant velocity in the x and y dimensions respectively. v_x and v_y represent the number of pixels the target traverses per frame and do not need to be integer values. Consider the case where N_F filters are used and filtering begins at time $n_t = 0$. Taking the z-transform of Equation 3-4 gives

$$S(z_x, z_y, z_t) = \sum_{n_x=-\infty}^{\infty} \sum_{n_y=-\infty}^{\infty} \sum_{n_t=0}^{N_F-1} s(n_x - v_x n_t, n_y - v_y n_t) z_x^{-n_x} z_y^{-n_y} z_t^{-n_t} \qquad \text{eq 3-5}$$

Defining $m_x = n_x - v_x n_t$ and $m_y = n_y - v_y n_t$, yields

$$S(z_x, z_y, z_t) = \sum_{m_x=-\infty}^{\infty} \sum_{m_y=-\infty}^{\infty} \sum_{n_t=0}^{N_F-1} s(m_x, m_y) z_x^{-(m_x + v_x n_t)} z_y^{-(m_y + v_y n_t)} z_t^{-n_t} \qquad \text{eq 3-6}$$

Rearranging

$$S(z_x, z_y, z_t) = \sum_{n_t=0}^{N_F-1} \left[\sum_{m_x=-\infty}^{\infty} \sum_{m_y=-\infty}^{\infty} s(m_x, m_y) z_x^{-m_x} z_y^{-m_y} \right] z_x^{-v_x n_t} z_y^{-v_y n_t} z_t^{-n_t} \qquad \text{eq 3-7}$$

Noting that the bracketed term is in fact the z-transform of the spatial components of the spatio-temporal signal, reduces Equation 3-7 to

$$S(z_x, z_y, z_t) = \sum_{n_t=0}^{N_F-1} S(z_x, z_y) z_x^{-v_x n_t} z_y^{-v_y n_t} z_t^{-n_t} \qquad \text{eq 3-8}$$

The conjugate of the above expression is used in the expression for the matched filter,

$$S(z_x^{-1}, z_y^{-1}, z_t^{-1}) = S(z_x^{-1}, z_y^{-1}) \sum_{k_t=0}^{N_F-1} z_x^{v_x k_t} z_y^{v_y k_t} z_t^{k_t} \qquad \text{eq 3-9}$$

Substituting the above expression into the expression for the matched filter (Equation 3-3) yields

$$H(z_x, z_y, z_t) = \frac{S(z_x^{-1}, z_y^{-1})}{\Sigma(z_x, z_y, z_t)} \left[\sum_{k_t=0}^{N_F-1} z_x^{v_x k_t} z_y^{v_y k_t} z_t^{k_t} \right] \qquad \text{eq 3-10}$$

where the bracketed term is the transfer function of the velocity filter. The impulse response of the velocity filter is given by

$$h_\upsilon(n_x, n_y, n_t) = \sum_{k_t=0}^{N_F-1} \delta\left(n_x + v_x k_t, n_y + v_y k_t, n_t + k_t\right) \qquad \text{eq 3-11}$$

where n_x ,n_y and n_t take values less than or equal to zero. Note that the subscript υ refers to the particular velocity that the velocity filter is tuned to.

It is useful to consider an example to illustrate the technique described above. Consider the one dimensional case (one spatial dimension) where four frames are used and a velocity filter is employed that is tuned to a velocity of 1 pixel per frame. Thus

$$h_{\upsilon=1}(n_x, n_t) = \sum_{k_t=0}^{3} \delta\left(n_x + k_t, n_t + k_t\right) \qquad \text{eq 3-12}$$

Expanding the above yields

$$h_1(n_x, n_t) = \delta(n_x, n_t) + \delta(n_x + 1, n_t + 1) + \delta(n_x + 2, n_t + 2) + \delta(n_x + 3, n_t + 3) \qquad \text{eq 3-13}$$

Figure 3-1 shows a diagrammatic representation of the velocity filter for this example. Just as the 1D matched filter is the 'flipped' version of the signal, the velocity filter is also the flipped version of the signal. However the velocity filter is flipped in both spatial and temporal dimensions. Figure 3-2 shows the result of flipping the velocity filter in both dimensions. This velocity filter would be matched to a signal that begins in pixel one and has a velocity of one pixel per frame. If the velocity filter is swept across the frames it would be matched to signals that originate from different pixels.

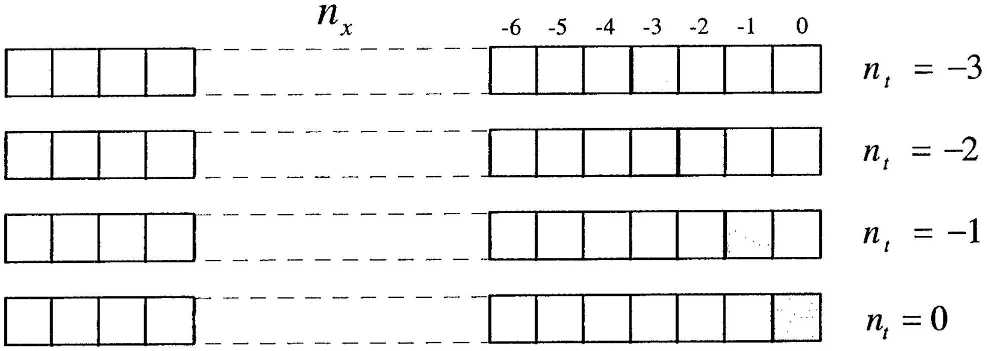


Figure 3-1: A diagrammatic representation of a one-dimensional velocity filter for an assumed velocity of one pixel per frame, over four frames.

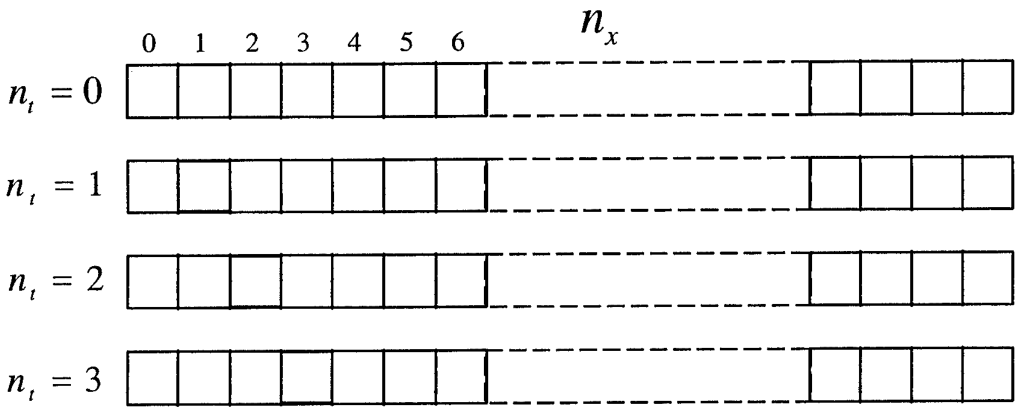


Figure 3-2: A' flipped' version of the velocity filter in the above example.

3.3 SNR Gain of the Velocity Filter

In this section the SNR gain of the velocity filter is derived. The results obtained here can be compared directly to the SNR gain of the matched filter derived in Section 2.4.

Consider a target located at (n_x, n_y, n_t). The SNR at the input to the velocity filter is given by

$$SNR_{IN} = \frac{s(n_x - v_x n_t, n_y - v_y n_t)^2}{\sigma^2} \qquad \text{eq 3-14}$$

where σ^2 is the variance of the background noise.

An expression for the SNR at the output of the velocity filter is now derived. Let $y = y_s + y_w$ be the velocity filter output where y_s and y_w are the signal and noise responses respectively. The output of the velocity filter is given by the convolution of the impulse response of the filter and the input to the filter. For the signal response

$$y_s(n_x, n_y, n_t) = s(n_x - v_x n_t, n_y - v_y n_t) * h(n_x, n_y, n_t) \qquad \text{eq 3-15}$$

Applying the expression for the velocity filter given in Equation 3-11 yields

$$\begin{aligned} y_s(n_x, n_y, n_t) &= \sum_{k_t=0}^{N_F-1} s(n_x - v_x n_t, n_y - v_y n_t) * \delta(n_x + v_x k_t, n_y + v_y k_t, n_t + k_t) \\ &= \sum_{k_t=0}^{N_F-1} s(n_x + v_x k_t - v_x(n_t + k_t), n_y + v_y k_t - v_y(n_t + k_t)) \\ &= \sum_{k_t=0}^{N_F-1} s(n_x - v_x n_t, n_y - v_y n_t) \qquad \text{eq 3-16} \end{aligned}$$

The above expression is independent of k_t thus

$$y_s(n_x, n_y, n_t) = N_F s(n_x - v_x n_t, n_y - v_y n_t) \qquad \text{eq 3-17}$$

Reference 4 shows that the output noise power can be expressed as

$$\sigma_{y_w}{}^2 = N_F \sigma^2 \qquad \text{eq 3-18}$$

This result occurs because the application of the velocity filter and the pre-processing that occurs prior to the velocity filter tend to decorrelate the background noise. Using the above results, The SNR at the output of the velocity filter is given by

$$\begin{aligned} SNR_{OUT} &= \frac{y_s{}^2(n_x, n_y, n_t)}{\sigma_{y_w}{}^2} = \frac{N_F{}^2 s(n_x - v_x n_t, n_y - v_y n_t)^2}{N_F \sigma^2} \\ &= \frac{N_F s(n_x - v_x n_t, n_y - v_y n_t)^2}{\sigma^2} \qquad \text{eq 3-19} \end{aligned}$$

The gain in terms of SNR achieved by using the velocity filter is thus

$$\begin{aligned} SNR_{Gain} &= \frac{SNR_{OUT}}{SNR_{IN}} \\ &= \frac{N_F s(n_x - v_x n_t, n_y - v_y n_t)^2}{\sigma^2} \Big/ \frac{s(n_x - v_x n_t, n_y - v_y n_t)^2}{\sigma^2} \\ &= N_F \qquad \text{eq 3-20} \end{aligned}$$

This result is analogous to the 1D matched filter case where a similar result for processing N samples instead of a single sample was obtained.

Table 3-1 summarizes the velocity filter and can be compared directly to Table 3-2, which summarizes the 1D and 3D matched filters. Table 3-1 shows both a one and two dimensional velocity filter. The one dimensional velocity filter only considers single dimensional frames, i.e. one spatial dimension and one time dimension. It is equivalent to a two dimensional matched filter. The two dimensional velocity filter considers two spatial dimensions and one time dimension. It is equivalent to a three dimensional matched filter.

Table 3-1: A comparison between the 1D and 2D velocity filters

	1D Velocity Filter	2D Velocity Filter
Measurement	$v(n_x, n_t)$	$v(n_x, n_y, n_t)$
Signal	$s(n_x, n_t) = s(n_x - v_x n_t)$	$s(n_x, n_y, n_t) = s(n_x - v_x n_t, n_y - v_y n_t)$
Noise	$w(n_x, n_t)$	$w(n_x, n_y, n_t)$
Noise variance (Input to filter)	σ^2	σ^2
Hypothesis, H_0	$v(n_x, n_t) = w(n_x, n_t)$	$v(n_x, n_y, n_t) = w(n_x, n_y, n_t)$
Hypothesis, H_1	$v(n_x, n_t) = s(n_x, n_t) + w(n_x, n_t)$	$v(n_x, n_y, n_t) = s(n_x, n_y, n_t) + w(n_x, n_y, n_t)$
Index	$\mathbf{n_x} = [n_x[0] \quad n_x[1] \quad \ldots \quad n_x[N_x - 1]]^T$ $\mathbf{n_t} = [n_t[0] \quad n_t[1] \quad \ldots \quad n_t[N_t - 1]]^T$	$\mathbf{n_x} = [n_x[0] \quad n_x[1] \quad \ldots \quad n_x[N_x - 1]]^T$ $\mathbf{n_y} = [n_y[0] \quad n_y[1] \quad \ldots \quad n_y[N_y - 1]]^T$ $\mathbf{n_t} = [n_t[0] \quad n_t[1] \quad \ldots \quad n_t[N_t - 1]]^T$
Filter	$h(n_x, n_t) = \sum_{k_t=0}^{N_F-1} \delta(n_x + v_x k_t, n_t + k_t)$	$h(n_x, n_y, n_t) = \sum_{k_t=0}^{N_F-1} \delta(n_x + v_x k_t, n_y + v_y k_t, n_t + k_t)$
Filter TF	$H(z_x, z_t) = \sum_{k_t=0}^{N_F-1} z_x^{-v_x k_t} z_t^{-v_t k_t}$	$H(z_x, z_y, z_t) = \sum_{k_t=0}^{N_F-1} z_x^{-v_x k_t} z_y^{-v_y k_t} z_t^{-v_t k_t}$
Output	$y = y_s + y_w$ y_s : target response, y_w : noise response $y_s(n_x, n_t) = s(n_x - v_x n_t) * h(n_x, n_t)$ $y_s(n_x, n_t) = N_F s(n_x - v_x n_t)$ $y_w(n_x, n_t) = w(n_x, n_t) * h(n_x, n_t)$ $y_w(n_x, n_t) = \sum_{k_t=0}^{N_F-1} w(n_x + v_x k_t, n_t + k_t)$	$y = y_s + y_w$ y_s : target response, y_w : noise response $y_s(n_x, n_y, n_t) = s(n_x - v_x n_t, n_y - v_y n_t) * h(n_x, n_y, n_t)$ $y_s(n_x, n_y, n_t) = N_F s(n_x - v_x n_t, n_y - v_y n_t)$ $y_w(n_x, n_y, n_t) = w(n_x, n_y, n_t) * h(n_x, n_y, n_t)$ $y_w(n_x, n_y, n_t) = \sum_{k_t=0}^{N_F-1} w(n_x + v_x k_t, n_y + v_y k_t, n_t + k_t)$
SNR	$SNR_{OUT} = \frac{N_F s(n_x - v_x n_t)^2}{\sigma_x^2}$	$SNR_{OUT} = \frac{N_F s(n_x - v_x n_t, n_y - v_y n_t)^2}{\sigma_x^2}$
SNR Gain	N_F	N_F

3.4 Performance of the Velocity Filter

Now that the SNR gain of the velocity filter has been derived the results can be used to assess the performance of the velocity filter. It is found that as for of the matched filter, the distributions of the noise and noise plus signal hypotheses along the test statistic for the velocity filter is dependent on the SNR.

A derivation of the probability of false alarm and probability of detection for a single velocity filter is provided in Reference 4. The probability of false alarm is the probability that a single application of a particular velocity filter will falsely report a target to be present. The probability of detection is the probability that a target at a particular location and travelling at a particular velocity will be detected by the appropriate velocity filter. In this section the results obtained from deriving the probability of false alarm and detection given in Section 2.5 are used to derive similar expressions for the velocity filter.

From (eq. 2-30) scaling the test statistic gives the following PDF for the matched filter

$$T' \sim \begin{cases} N(0,1) \mid H_0 \\ N(\sqrt{SNR_{OUT}},1) \mid H_1 \end{cases} \qquad \text{eq 3-21}$$

where the scaled test statistic is given by

$$T' = \frac{T}{\sqrt{\sigma^2 \varepsilon}} \qquad \text{eq 3-22}$$

If the velocity filter employs N_F frames, the SNR is increased by this factor. From Equation 3-20

$$SNR_{OUT} = N_F SNR_{IN} \qquad \text{eq 3-23}$$

Thus the scaled test statistic for the velocity filter gives the following PDF

$$T' \sim \begin{cases} N(0,1) \mid H_0 \\ N(\sqrt{N_F SNR_{IN}},1) \mid H_1 \end{cases} \qquad \text{eq 3-24}$$

where the scaled test statistic has become

$$T' = \frac{T}{\sqrt{N_F \sigma^2 \varepsilon}} \qquad \text{eq 3-25}$$

From Equation 2-35 the threshold for the test statistic for the matched filter is given by

$$\gamma' = \sqrt{\sigma^2 \varepsilon} Q^{-1}(P_{FA}) \qquad \text{eq 3-26}$$

Noting that the SNR input to the velocity filter is given by

$$SNR_{IN} = \frac{\varepsilon}{\sigma^2}$$

and also using Equation 3-20, the threshold of the test statistic for the velocity filter is given by

$$\gamma' = \sigma^2 \sqrt{N_F SNR_{IN}} Q^{-1}(P_{FA}) \qquad \text{eq 3-27}$$

Figure 3-3 shows the above results diagrammatically. It is clear to see the advantage of employing multiple frames. As the number of frames increases the PDF of the signal plus noise case moves to the right.

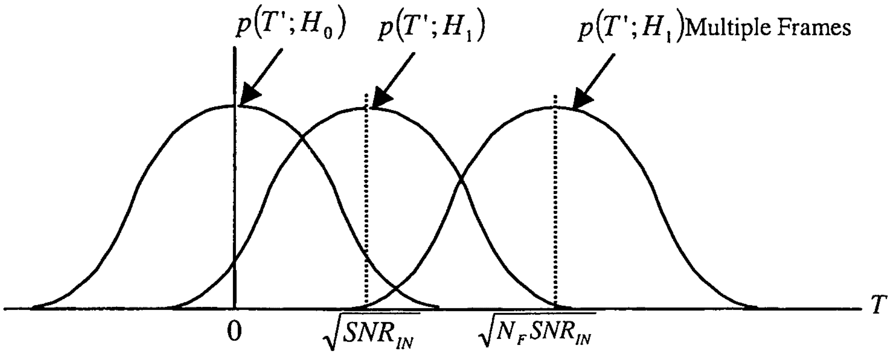


Figure 3-3: PDF's for the velocity filter

Using the above relationship between the number of frames and the SNR and combining this with Equation 2-36, leads to the following relationship between the probability of detection and the probability of false alarm.

$$P_D = Q\left(Q^{-1}(P_{FA}) - \sqrt{N_F SNR_{IN}}\right) \qquad \text{eq 3-28}$$

Figure 3-4 shows a theoretical Receiver Operating Characteristic (ROC) curve, which expresses Equation 3-28 over multiple probabilities of false alarm and numbers of filters. The ROC curve is a fundamental measure of performance that is commonly used. The SNR was set to four (6.02dB) for these simulations to match similar curves that are provided in Reference 4 for comparative purposes.

The figure below shows the improvement in performance obtained by increasing the number of frames. For a given false alarm probability, the probability of detection increases when additional frames are used.

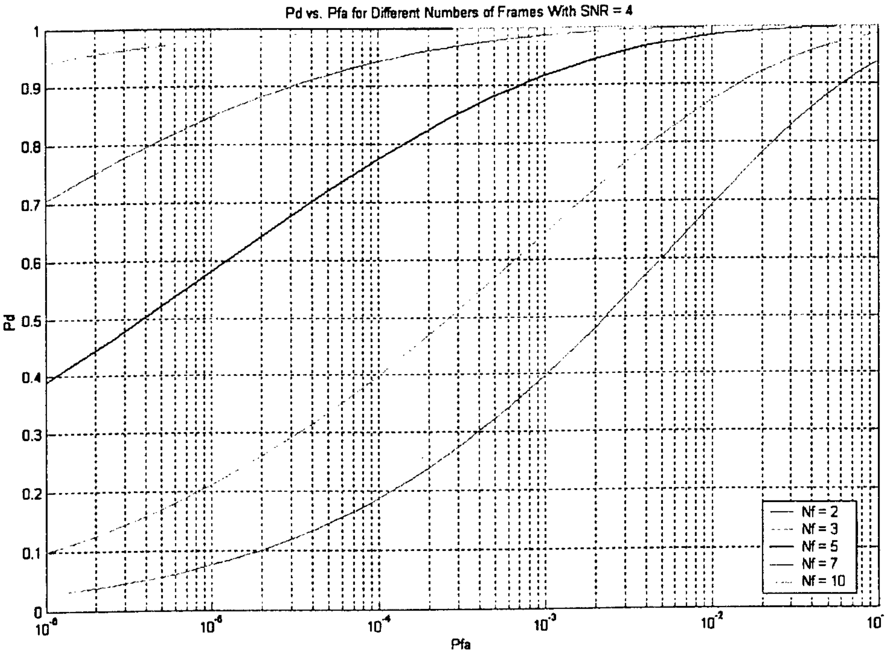


Figure 3-4: Theoretical velocity filter bank ROC curves

3.5 The Velocity Filter Bank

When the range of possible target velocities exceeds the width of a single velocity filter bin, more than one bin is required. The resulting collection of velocity filter bins forms a velocity filter bank. The number of bins used in the velocity filter bank is determined by the range of possible target velocities and the velocity range covered by a single bin.

In IR cases it is not the actual target velocity that the velocity filters are tuned to, rather it is the target velocity in pixels per frame. The target velocity in pixels per frame is a function of the actual target velocity relative to the detector and the distance between the detector and the target. Targets that are close to the detector will traverse more pixels per frame than those further away. Thus in determining the range of velocities in pixels per frame that the velocity filter bank should cover, it is not only the actual target velocity relative to the detector but also the expected range to the target that must be considered.

It is important to note that the output of the velocity filter bank tells the user which pixel the target started in and its velocity in pixels per frame. The target could be moving towards or away from the sensor and still fall into the same velocity filter bin. Some type of secondary processing would need to be performed if the target velocity vector needed to be found. For example a rough estimate of the targets velocity vector could be obtained by noting the rate of change of the SNR from the target.

3.5.1 Calculating the Maximum Number of Velocity Filters & the Velocity Filter Bank Resolution for a 1D Velocity Filter

Assume that targets can move between $v_{\min}$ and $v_{\max}$ pixels per frame. Hence, after n frames the target can move into any of $(n-1)(v_{\max}-v_{\min})+1$ pixels. Thus, there are $(n-1)(v_{\max}-v_{\min})+1$ possible target velocities and hence velocity filters

$$N_{VF}=(n-1)(v_{\max}-v_{\min})+1 \qquad \text{eq 3-29}$$

where N_{VF} is the number of velocity filters.

The distance in pixels per frame between adjacent members of the velocity filter bank is termed the resolution of the velocity filter bank in this report. The resolution is inversely proportional to the number of frames.

$$\Delta_{VF}=\left(\frac{1}{n-1}\right) \qquad \text{eq 3-30}$$

Figure 3-5 shows an example of the above. In this figure it is assumed that the target can have velocities ranging between -1 and 1 pixels per frame. This is shown by the shaded pixels. Each line passing through the frames corresponds to a particular target velocity. The region between each line represents the resolution of the velocity filter bank. As the number of frames is increased it is clear to see that the number of velocity filters increases and the resolution improves. Thus a finer resolution in velocity measurements is obtained at the expense of an increased number of computations and increased latency before a detection is declared.

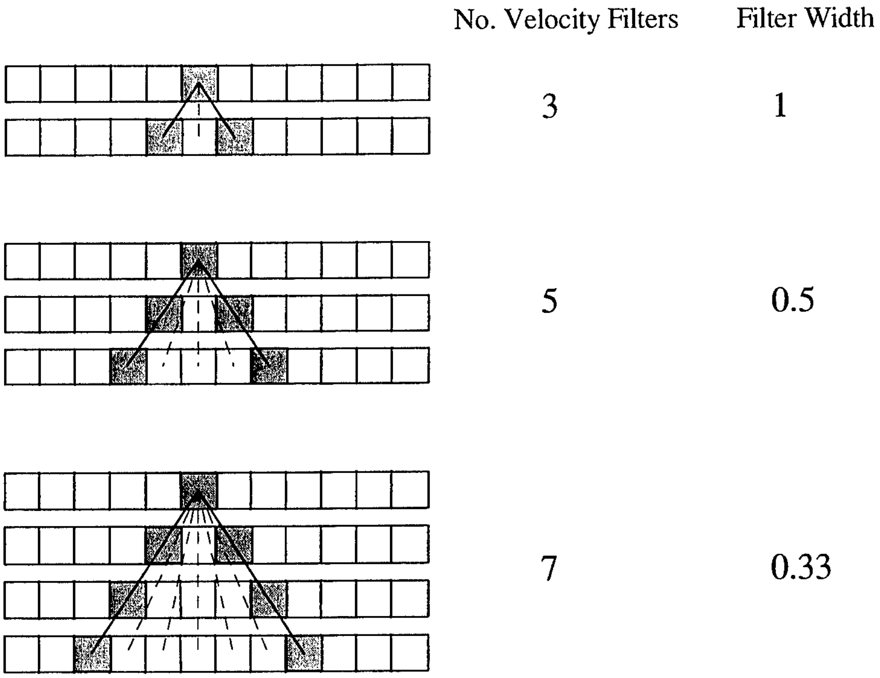


Figure 3-5: The number of velocity filters grows with the number of frames

It is important to note that the above calculations refer to the maximum number of velocity filters and the highest possible resolution of the velocity filter bank. Depending on a number of issues including computational costs, the expected width of the target signature in velocity and the required update rate, the user may not choose to implement all possible velocity filters.

The results presented in this section apply to the one dimensional velocity filter. The results have shown that the number of velocity filters increases linearly with the number of frames. For a two dimensional case the number of possible velocity filters increases at a rate proportional to the square of the number of frames. It is clear to see that after only a few frames the number of possible velocity filters gets very large. Thus it is particularly important to prune the number of velocity filters used in the 2D case. One solution is to omit every n^{th} velocity filter. This reduces the number of filters at the expense of a reduction in the resolution. Reference 7 presents an alternative solution whereby 2D space is projected into 1D space before applying the velocity filter. This reduces the number of computations at the expense of a reduction in performance. Several other methods used to reduce the number of computations are presented in the next section.

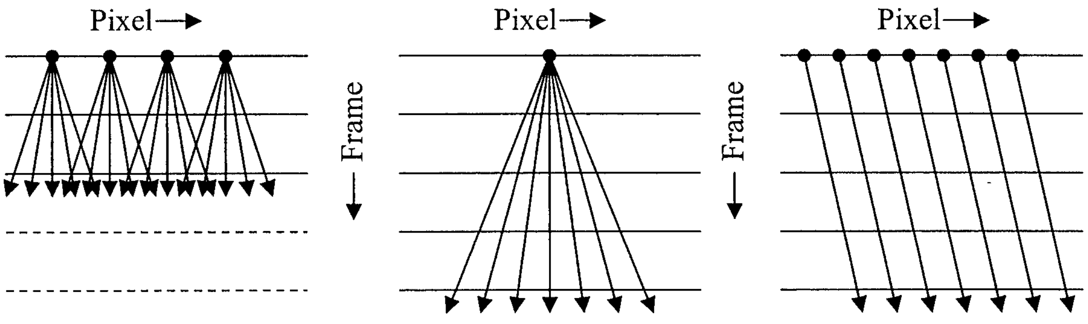
3.6 Velocity Filter Implementation Issues

In the preceding sections the velocity filter has been developed and the concept of the velocity filter bank has been discussed. This section discusses several velocity filter implementation issues. Figure 3-6 shows three velocity filter computation strategies. Each horizontal line in the figure corresponds to a frame and the lines passing through these frames correspond to the velocity filters. The difference between the three techniques is the order in which the filtering is performed.

Figure 3-6(a) shows a frame-first strategy. Each velocity filter state in each pixel is updated before moving onto the next frame. Using this strategy the velocity filter states (or partial sums) for each pixel in the current frame must be stored in memory. The frames on the other hand do not have to be stored. The advantage of the frame-first strategy is that it lends itself to recursive velocity filtering. Using recursive velocity filtering, updates to the velocity filter states are obtained after each frame. Thus the entire frame stack does not need to be processed before a target can be declared.

There are two main disadvantages of employing the frame-first technique. In many applications there are hundreds or thousands of velocity filters but only a few tens of frames. Thus storing the frames requires far less memory than storing the filters. Secondly, the temporal correlation of the false alarms is much higher for a frame-first implementation than a pixel-first or filter-first implementation [4].

The pixel-first and filter-first implementations, shown in Figure 3-6(b) and (c), store the frames, not the velocity filters states for each pixel. In the pixel-first implementation, the filter trajectories starting at a given pixel in the first frame are traced all the way to the bottom of the frame stack. This process is repeated for all pixels in the first frame. In the filter-first implementation, a single filter is applied to all pixels in the first frame and is traced all the way to the bottom of the stack. The filter is then changed and the process repeated until all filters in the velocity filter bank have been applied to the stack.



(a) Frame-First (b) Pixel-First (c) Filter-First

Figure 3-6: Velocity filter computation strategies

3.6.1 Sequential Velocity Filter

The disadvantage of the pixel-first and filter-first techniques is that it is not until all the frames have been processed that decisions can be made as to the presence of a target. Reference 8 develops a modification on the basic velocity filter, which addresses this problem. Unlike the basic velocity filter, which delays thresholding until the entire frame has been processed, the sequential velocity filter applies thresholding after each frame. If the integrated sum is greater than the upper threshold, a target is declared and further processing is halted. If the sum falls beneath a lower threshold, it is determined that no target is present for that filter at that location and once again processing is halted. The above process is illustrated in Figure 3-7.

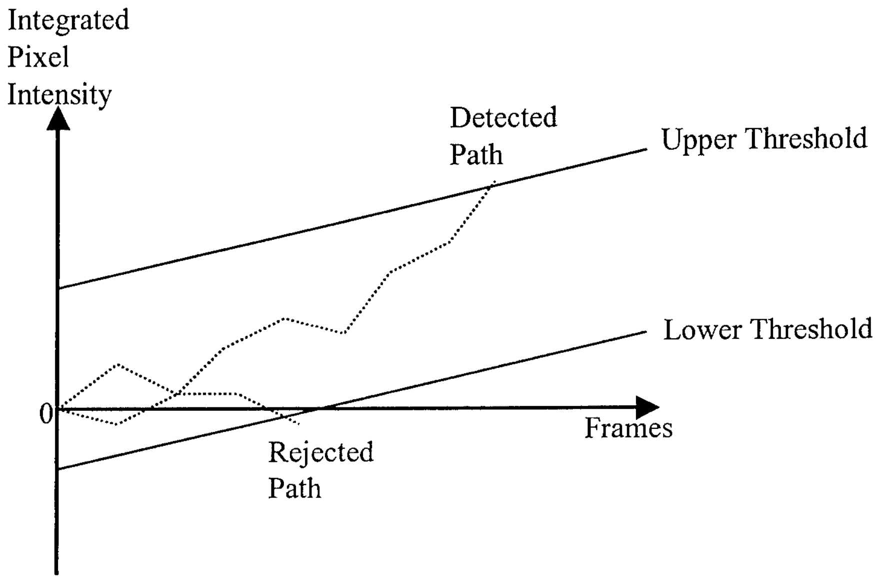


Figure 3-7: The sequential velocity filter

The sequential velocity filter is most effectively implemented using the pixel-first approach. Using this approach allows the exploitation of cases where adjacent filters share the same quantised path for the first several frames before diverging.

3.6.2 Single Bit Velocity Filter

References 3 and 6 discuss the use of a single bit velocity filter. The single bit velocity filter quantises the data into two levels. This substantially reduces the memory requirements at the expense of a reduction in performance. The single bit velocity filter requires two thresholds. The first is used to determine which pixels in the input data will be given a value of one i.e. the maximum or 'on' value. The velocity filter bank is then applied to the quantised image and a count is kept of the number of times a level of one was observed in the quantised image for each application of each velocity filter. The output of the filter bank is then processed and a target will be declared for a particular velocity filter and initial target location if the count is above a second threshold.

3.6.3 Implementing the 1D Matched Filter & the 1D Velocity Filter: A Comparison

Table 3-2 compares the 1D matched filter with the 1D velocity filter in terms of implementation. As defined early in this chapter, the 1D velocity filter has a single spatial dimension and a time dimension. The aim of this section is to assist the reader in understanding the implementation of the velocity filter by relating it to that of the matched filter.

For the 1D matched filter, convolution is performed in the domain in which there is uncertainty about the target, i.e. the time domain. The convolution is equivalent to flipping the filter and moving along the measurement vector from left to right. Theoretically the maximum output will occur when the filter exactly overlaps the signal. This allows the location of the signal in time to be determined. The same principle applies for the 1D velocity filter. For the 1D velocity filter the uncertain terms are the initial pixel location of the target and the velocity of the target. These are the dimensions in which convolution must occur. Each velocity filter is flipped both horizontally (along the pixels) and vertically (along the frames) and shifted from left to right over the pixels. Theoretically the maximum output will occur when the appropriate velocity filter exactly overlaps the signal. Thus the velocity of the signal and its initial pixel position can be determined. Note that there is no need to convolve in the time domain, as one of the assumptions of the velocity filter is that if a signal is present it will occupy all frames.

Table 3-2: A comparison between the implementation of the 1D matched filter and the 1D velocity filter

	1D Matched Filter	1D Velocity Filter
Known terms	Form of the signal • Shape • Energy	Form of the signal • Shape • Energy • Signal is located in every frame • Signal structure remains constant over frames
Unknown terms	Location of the signal in time (or sample number)	Location of the signal (initial pixel location) Velocity of the signal
Convolution	Time domain	Spatial domain (along the pixels) Velocity domain
Principle	Maximum output occurs when the filter and signal overlap	Maximum output occurs when the filter and signal overlap
Implementation method	1. Flip the filter in time 2. Move the filter from left to right over the measurement summing the response at each point 3. Threshold the output 4. Declare elements that pass the threshold, detections	1. Flip the filter both horizontally (pixels) and vertically (frames) 2. Shift the filter from left to right over the pixels summing the response at each point (filter-first implementation) 3. Repeat for the next member of the velocity filter bank 4. Threshold the output 5. Declare elements that pass the threshold, detections

4. Velocity Filter Simulations

This section discusses simulations that were performed to test the performance of a 1D velocity filter. A decision was made to implement the 1D velocity filter, as it requires far less computation time and is easier to visualise that the 2D velocity filter. Due to the time constraints imposed on the project, stringent assumptions needed to be made. The assumptions made and parameters used for the tests were:

- The target occupies only one pixel at a time
- The target can only move a maximum of one pixel per frame
- The noise was zero mean white Gaussian noise with a variance equal to one
    - No temporal correlation in the noise
    - No spatial correlation in the noise (independent from pixel to pixel in a particular frame)
- There were a total of 50 pixels in each frame.

4.1 Types of Simulations Performed

This section begins with the results from simulating the velocity filter technique for a single target. Following this the same simulations were performed for two targets. In each of the above simulations it was assumed that the noise variance was equal to one. This is an important parameter as the value of the noise variance is used in determining a threshold for detections (see Equation 3-27). In Section 4.1.3 a technique is developed to estimate the noise variance. The aim is to determine the effect that using an estimated noise variance value has on the simulations. Such simulations were performed in Section 4.1.4 where ROC curves were developed for the velocity filter technique. In all cases the filter-first technique as discussed in Section 3.6 was implemented.

4.1.1 Single Target

In the simplest case the velocity filter technique was implemented for the case where a single target is present and an assumption is made that the noise variance is equal to one. In the figures that follow the probability of false alarm was set to 10^{-4}, ten frames were used and the SNR was set to 3dB. Figure 4-1 shows the input signal and the signal plus noise which is the input to the velocity filters. The target starts in pixel ten and moves at a velocity of 0.5 pixels per frame. The low SNR makes the signal difficult to detect over the background clutter.

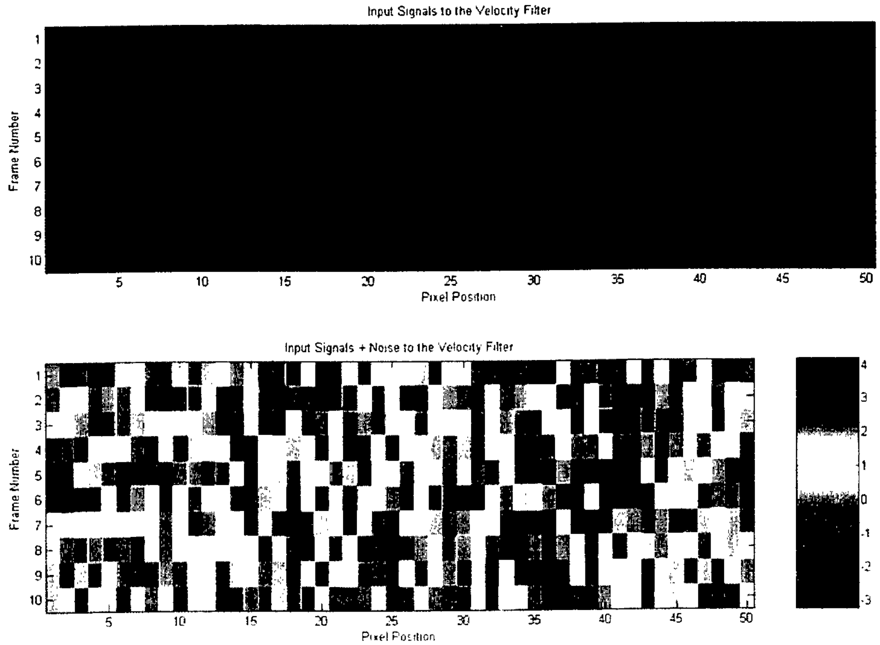


Figure 4-1: Input signal and noise into the velocity filter

As discussed in Section 3.6.3 the output of the velocity filter is the convolution of the input to the filter and the space temporal form of the filter. Theoretically the maximum output occurs when the velocity filter matches the target exactly in both velocity and initial pixel position. Figure 4-2 shows the two dimensional convolution between the velocity filter and the input to the filter for three different velocity filters. In the top plot the velocity filter exactly matched the actual target velocity. The diagonal line of maximum values is representative of the target motion over multiple frames. In the other plots the output is not as large as the filter is not matched perfectly to the target (note the changing scale between the three plots). If a target exists, it will be present in all frames thus it is unnecessary to convolve in the frame (or time) dimension. Only the case where the filter perfectly overlaps the input signal needs to be considered. This occurs along the central horizontal line of each plot. Thus only this central line needs to be kept in memory. In fact only the elements of this line that correspond to those where non zero elements of the filter overlap the input image, need to be kept in memory, i.e. the blank sections shown on the right hand side of the plots in Figure 4-2 can be discarded.

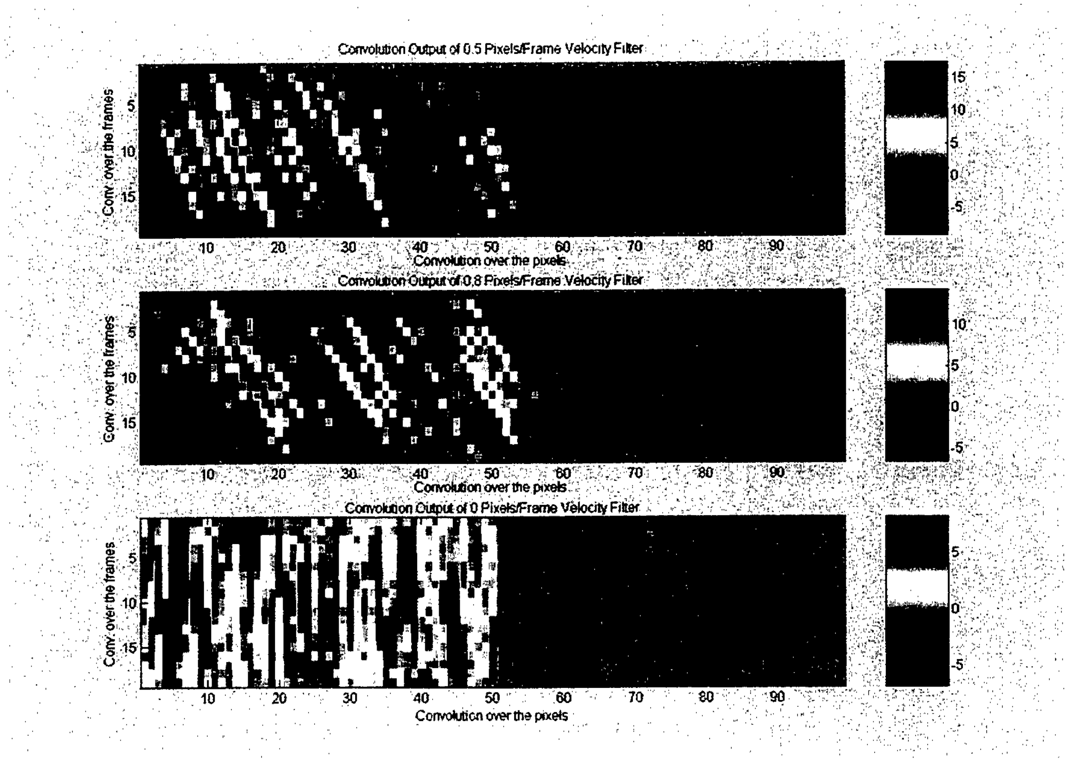


Figure 4-2: Convolution output for various velocity filters

The purpose of the above discussion is to assist in the understanding of Figure 4-3. Simulations were not performed using convolutions; rather the implementation method discussed in Table 3-2 was used[1]. The top plot in Figure 4-3 shows the output of the velocity filter bank for the input illustrated in Figure 4-1. A total of 19 velocity filters were used. These filters were effectively moved from left to right over the input frames. Each cell for the top plot in Figure 4-3 represents the output of a particular velocity filter assuming that the target starts at a particular initial pixel position. Each horizontal line corresponds to the output for a particular velocity filter applied to all the pixels. This is equivalent to a section of the central horizontal line shown in the top diagram in Figure 4-2.

A threshold is applied to the output of the velocity filter and those elements that pass the threshold are declared targets. The central plot of Figure 4-3 shows the elements that have passed the threshold. Only one such element exists, which corresponds to a target with a velocity of 0.5 pixels per frame, starting in pixel ten. Thus, the velocity filter has correctly detected the targets velocity and initial starting position. The bottom plot of Figure 4-3 shows the trajectories for those elements that pass the threshold test.

[1] In practice, the implementation represented a correlation between the velocity filter and the input frames, not a convolution. The correlation process produces an equivalent result to the convolution process, as the velocity filter has uniform values over all frames.

This plot is equivalent to the input signal to the velocity filter shown in the top plot of Figure 4-1.

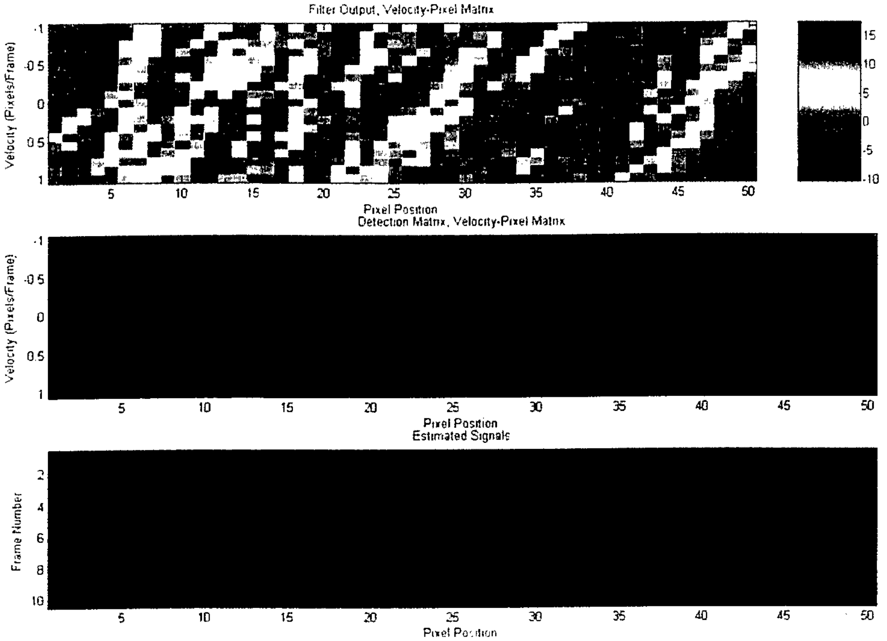


Figure 4-3: Output of the velocity filter bank

4.1.2 Multiple Target Detection Using Single Target Velocity Filters

An advantage of the velocity filter is that the number of computations performed is independent of the number of targets present. An example of the velocity filter technique applied to a case where the input contains multiple targets is shown in Figure 4-4. There are two targets present in this example. The first is the same target discussed in the previous section. The second is a target that has a velocity of -0.8 pixels per frame and begins in pixel 35. The SNR of both targets was 3dB, the false alarm rate was set to 10^{-4} and the number of frames was set to ten. Figure 4-5 shows the output of the velocity filter bank. By comparing the bottom plot in Figure 4-5 with the top plot of Figure 4-4 it is clear that both targets have been detected.

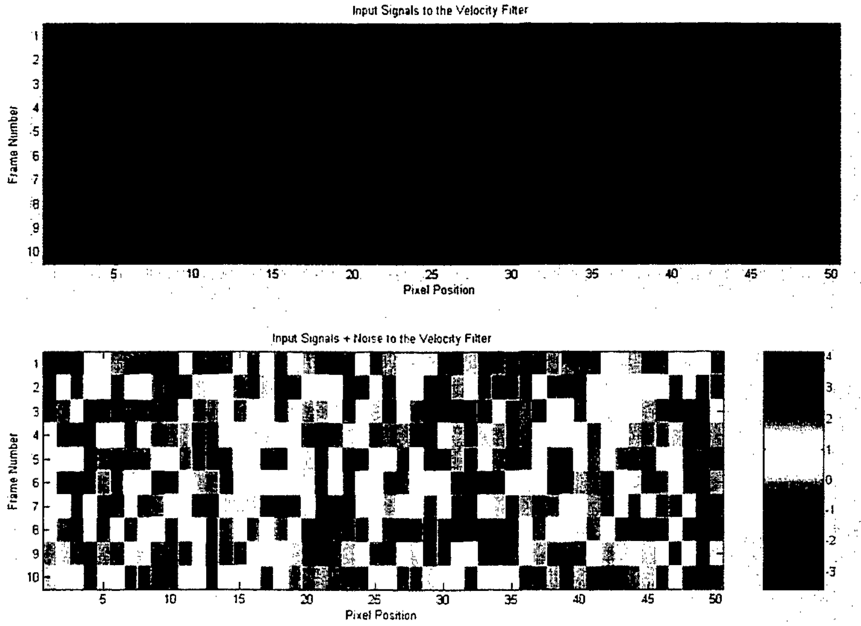


Figure 4-4: Input signals and noise into the velocity filter bank for multiple targets

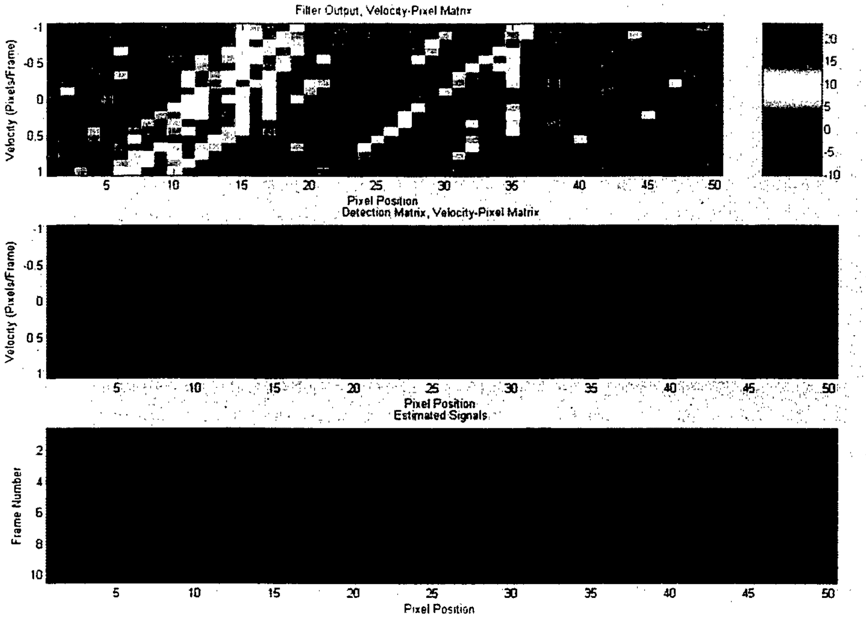


Figure 4-5: Output of the velocity filter bank for multiple targets

Often multiple elements corresponding to the same target may pass the threshold test at the output of the velocity filter bank. Figure 4-6 illustrates such a case[2]. In this figure a total of five elements have passed the threshold test. Clearly two of these detections are for the first target and three are for the second. In some cases the targets may be spaced closely together, for example aircraft flying in formation. A problem exists for these cases in associating the detection with the correct target and determining how many targets are present.

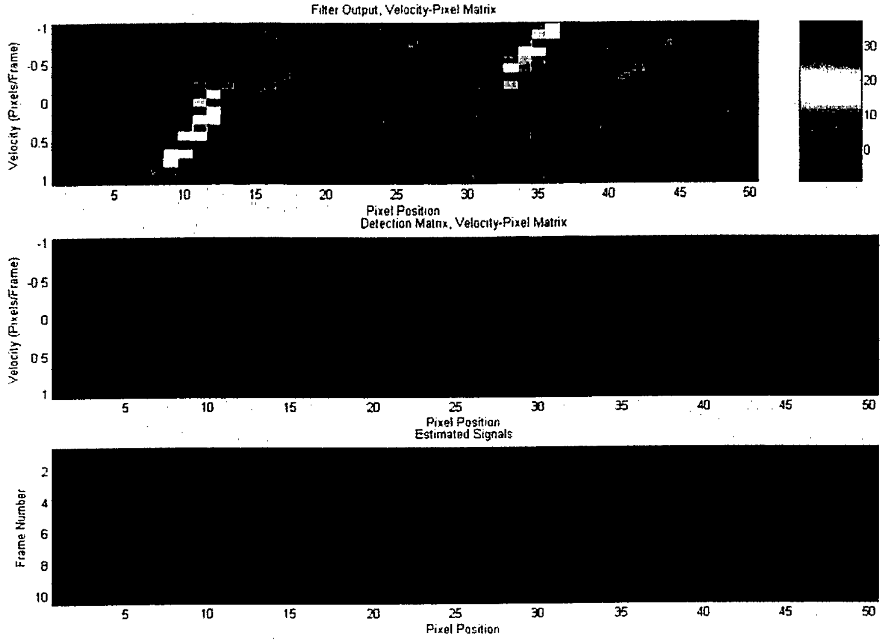


Figure 4-6: Multiple detections of multiple targets

4.1.3 Noise Variance Estimation

In the previous sections it was assumed that there was prior knowledge of the noise variance. Unless pre-processing of the data has occurred that has effectively set the noise variance of the data to a specific value, this will not be the case. Thus the noise variance must be determined via processing the data input to the velocity filter bank. In this section a recursive technique for estimating the noise variance is developed.

[2] Note that in Figure 4-6 the SNR of both targets is 5dB.

The variance, σ^2 of a set of data, $y = [y_1 \quad y_2 \quad \dots \quad y_n]$ is given by Reference 9

$$\overline{\sigma^2} = \frac{1}{n-1}\sum_{j=1}^{n}\left(y_j - \overline{y}\right)^2 \qquad \text{eq 4-1}$$

Assuming that the noise is zero mean white Gaussian, Equation 4-1 can be simplified to

$$\overline{\sigma^2} = \frac{1}{n-1}\sum_{j=1}^{n} {y_j}^2 \qquad \text{eq 4-2}$$

The difficulty with directly applying this formula to the data is that samples that contain the signal as well as the noise will be used in the calculations of the noise variance. However, without having prior knowledge of the noise variance a threshold level cannot be used to determine which output samples contain signal components. This is a fundamental problem in estimation. To overcome this problem a recursive algorithm was used to estimate the noise floor. Consider Equation 4-2 for the case where n-1 samples have been used to estimate the noise variance, i.e.

$$\overline{\sigma_{n-1}^2} = \frac{1}{(n-2)}\sum_{j=1}^{n-1} x_j^2 = \frac{1}{(n-2)}\left(x_1^2 + x_2^2 + \dots + x_{n-1}^2\right) \qquad \text{eq 4-3}$$

Now consider the case where n samples are used for the noise variance estimation

$$\overline{\sigma_n^2} = \frac{1}{(n-1)}\sum_{j=1}^{n} x_j^2 = \frac{1}{(n-1)}\left(x_1^2 + x_2^2 + \dots + x_{n-1}^2 + x_n^2\right) \qquad \text{eq 4-4}$$

Combining the results from the above equations yields

$$\overline{\sigma_n^2} = \left(\frac{1}{n-1}\right)\left((n-2)\overline{\sigma_{n-1}^2} + x_n^2\right) \qquad \text{eq 4-5}$$

This recursive form of the noise variance estimation allows each sample to be tested prior to being included in the noise variance estimate. If the value of the sample is determined to be too large a decision is made that the sample is likely to contain signal components and therefore will not be included in the noise variance estimate. The difficulty then lies in determining a threshold for deciding if a sample contains a signal. If the threshold is set too high, some samples containing signal components will be included in the estimate and bias the estimate to a high value. If the threshold is too low some samples consisting only of noise will not be included in the noise estimate and hence bias the estimate to a low value. In the simulations performed in this study, trial and error showed that setting the threshold to a value of twice the standard deviation of the noise estimate provided a good compromise. In practice a less ad-hoc method should be used to determine the threshold.

4.1.4 Simulated ROC Curves

This section shows the performance of the velocity filter bank using simulated ROC curves. It is important to note that the ROC curves presented here are only theoretical. An ROC curve plots the probability of detection versus the probability of false alarm, often for different variables such as the SNR or the number of frames. To create an ROC curve from simulations one would run the program multiple times, count the number of false alarms and the number of detections and use this information to create the plot. In this project, such an approach was not feasible as the required ROC curve includes probability of false alarm rates as low as one in a million. The code would therefore need to be executed on average one million times to count just a single false alarm. Techniques such as 'Importance Sampling' are beyond the scope of this report. In this study, the approach used to construct the ROC curves was to set a constant false alarm rate (CFAR)[3] and to use this to determine the detection threshold. Determining the probability of detection was then simply a matter of dividing the number of detections by the total number of runs. This produced a curve of probability of detection versus constant false alarm rate.

[3] In determining the CFAR it was noted that the probability of false alarm increases linearly with the number of frames [4]. Thus, it was important to scale the false alarm rate used in determining the threshold for detections.

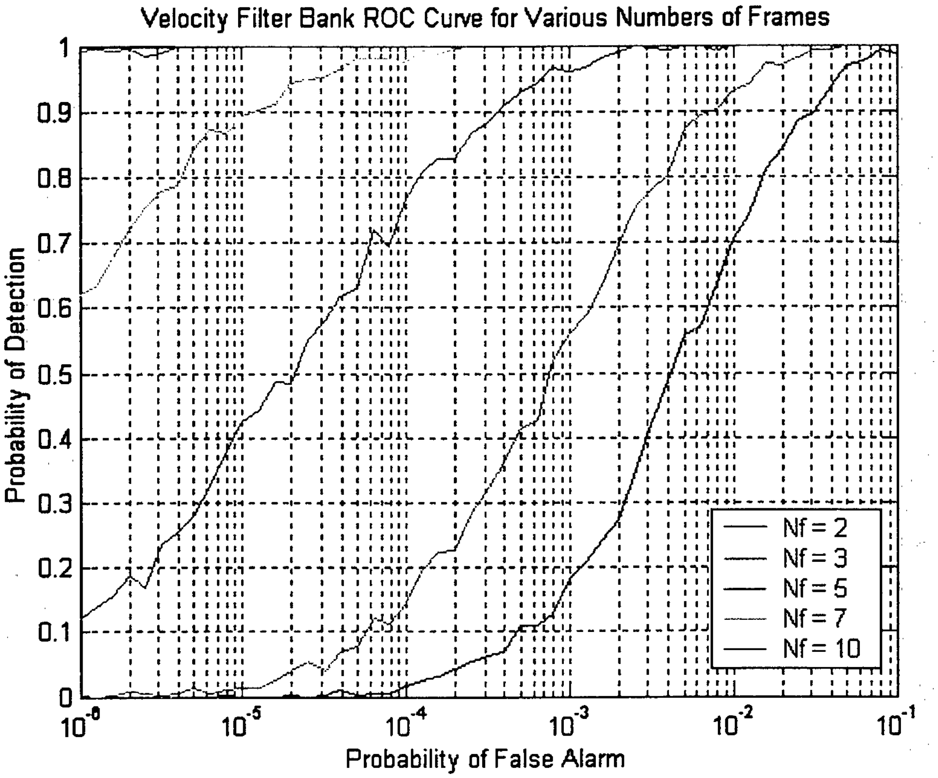


Figure 4-7: Velocity filter bank ROC curves for various numbers of frames using the true noise variance

Figure 4-7 shows several ROC curves for different numbers of frames. The true noise variance was used in the production of these curves. The SNR was set to 4 (6.02 dB), to match the simulations performed in Chapter 3. The advantage of using additional frames may be clearly seen. For a given probability of false alarm, the probability of detection substantially increases when the number of frames is increased.

Comparing Figure 4-7 with the corresponding theoretical plot from Chapter 3, Figure 3-4, shows that in general the simulated performance agrees with the theoretical performance. Both plots clearly show the improvements in probability of detection obtained by employing additional frames. However there are some notable differences between the plots. For probability of detection values above approximately 0.7 the two figures are similar however the performance of the simulated velocity filter decreases for lower values of probability of detection compared to the theoretical plots. For each probability of false alarm point in Figure 4-7, 500 simulations were performed. Performing such a large number of simulations means that random affects are unlikely to be the cause of the differences between the theoretical and simulated results. These differences should be investigated further if additional work is undertaken in this area.

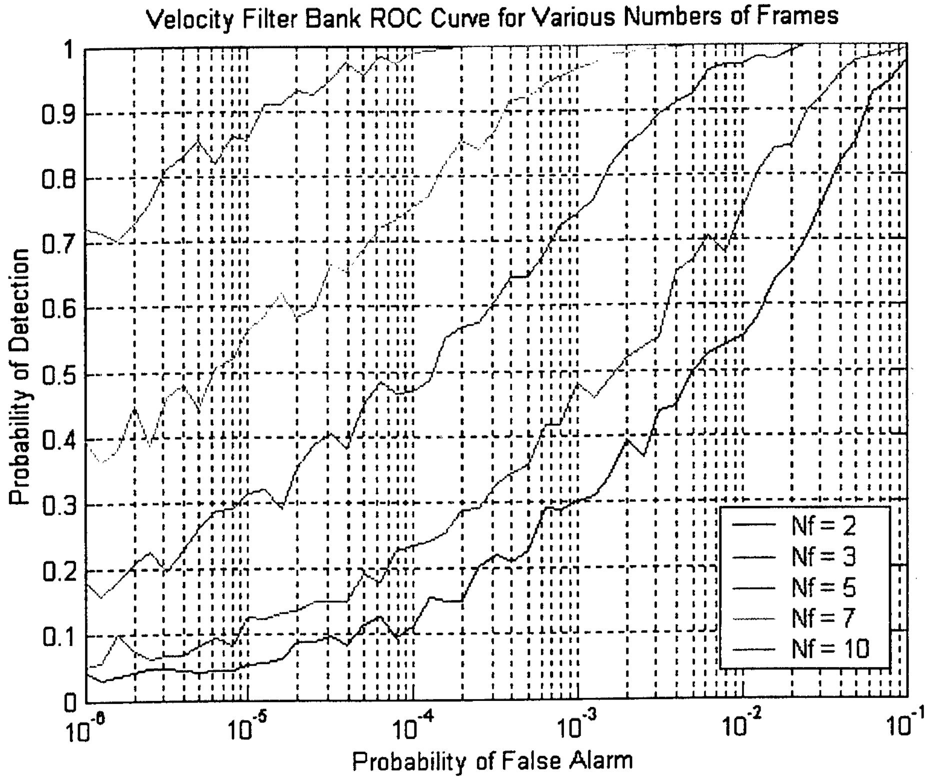


Figure 4-8: Velocity filter bank ROC curves for various numbers of frames using an estimated noise variance

The same simulations discussed above were rerun using the noise variance estimate. The results are shown in Figure 4-8. The slopes of the curves appear steeper when the noise variance was known. A reasonable explanation for these differences can be found by considering the uncertainty of the noise variance estimation. For each run the results of the noise variance estimation process will not be the same. For some runs the estimate will be less than the real value and in others it will be greater. In the cases where the noise variance estimate is less than the true value, more detections will be declared than if the true value was used. This will become more noticeable when the expected probability of detection from Figure 4-7 is very low. Similarly, when the expected probability of detection is very high there will be many cases where the noise variance estimate is a high value and some targets will not pass the threshold. The net result of the above is a smoothing in the ROC curves. Further investigations would need to be performed to confirm these results and the reasoning for the differences between Figures 4-7 and 4-8.

Figures 4-9 and 4-10 show a set of ROC curves for different SNR. Five frames were used in each simulation. In Figure 4-9 the true value of the noise variance was used where as an estimated value was used in Figure 4-10. Both figures show the improved performance of the velocity filter when the SNR is increased. The figures also show that velocity filtering works well in low SNR environments. It is in these environments that the velocity filter is most likely to be employed, as single measurements will detect the target when the SNR is large, and there will be little to gain by integrating the frames.

The ROC curves in Figure 4-9 show an improved performance over the curves in Figure 4-10. For example, consider the case where the SNR is 5dB and the probability of false alarm is set to 10^{-3}. The corresponding probability of detection is approximately 0.77 when the true noise variance is used compared to a value of approximately 0.54 when an estimate is used for the noise variance. As the threshold for declaring detections is directly proportional to the noise variance (see Equation 3-27) the above results indicate that the average of the noise variance estimates was greater than the true value. Thus the threshold for declaring whether a measurement contains noise only or signal plus noise was probably set too low.

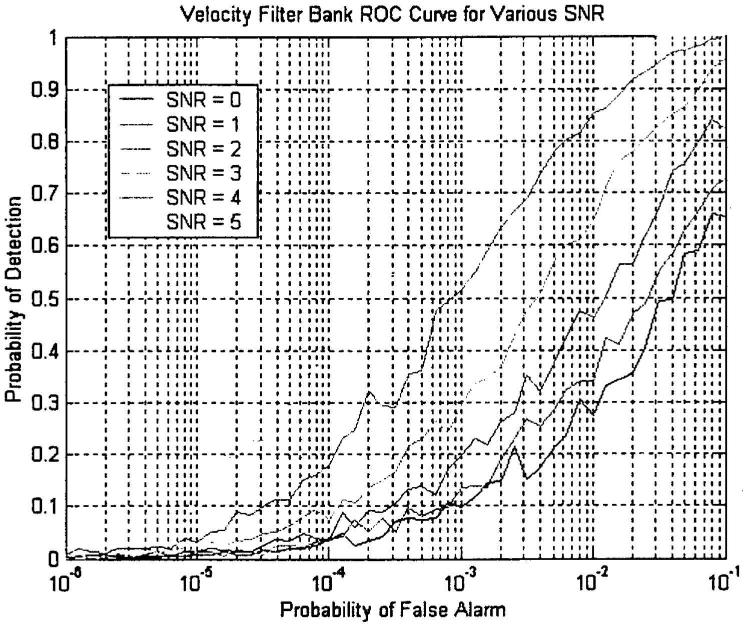


Figure 4-9: Velocity filter bank ROC curves for various SNR (dB) values: true noise variance

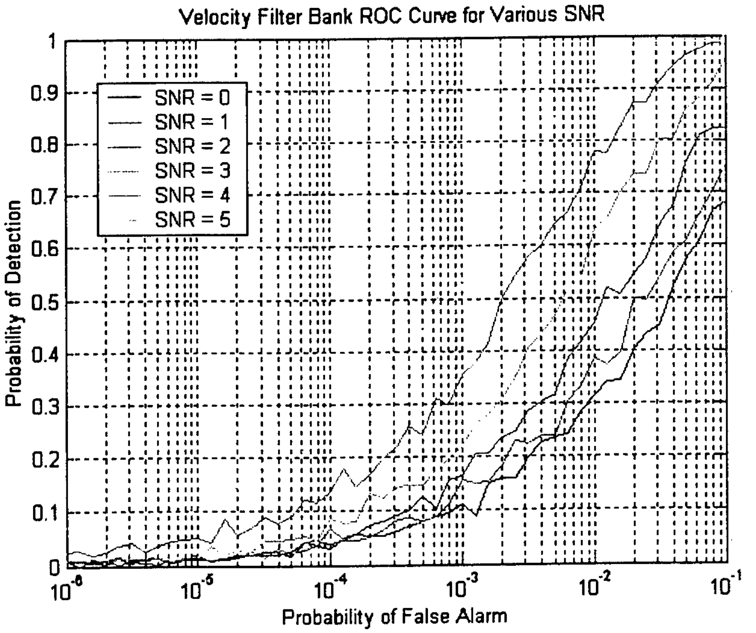


Figure 4-10: Velocity filter bank ROC curves for various SNR (dB) values: estimated noise variance

5. Conclusions & Recommendations for Further Work

The velocity filter technique is a particular type of track before detect technique where it is assumed that the target velocity is constant over the period that the frames are being integrated. In this report the velocity filter has been derived and simulated results have been compared to theory.

The report has shown the benefit of using the velocity filter in low SNR environments. When multiple frames are combined and the appropriate velocity filter is used, the signal components will add coherently. The noise components will not add coherently and an overall SNR gain will be achieved.

The report began with the derivation of the 1D matched filter. The test statistic for the 1D matched filter was derived, as was its distribution. It was found that the larger the signal to noise ratio, the further apart the noise only and the signal plus noise distributions will be along the test statistic (see Figure 2-3).

In Chapter 3 the velocity filter was derived. The impulse response of the velocity filter is given by

$$h_v(n_x, n_y, n_t) = \sum_{k_t=0}^{N_F-1} \delta\left(n_x + v_x k_t, n_y + v_y k_t, n_t + k_t\right) \qquad \text{eq 5-1}$$

When the uncertainty in the target velocity is such that the width of a single velocity filter is not large enough to cover this uncertainty a set of velocity filters is required. This is termed the velocity filter bank.

It was found that increasing the number of frames over which the velocity filters are applied had the following effects

- An increased SNR out of the velocity filter, proportional to the number of frames.
- An increase in the number of possible velocity filters, proportional to the number of frames for the 1D velocity filter and proportional to the square of the number of frames for the 2D velocity filter.
- An increase in the resolution of the velocity measurement, proportional to the number of frames.
- The probability of detection increases at a rate greater than the probability of false alarm, thus there is an overall improvement achieved by increasing the number of frames.
- Increase in the delay before track initiation can take place.

Three different techniques for implementing the velocity filter were presented, namely the frame-first, pixel-first and filter-first techniques. For the pixel-first and filter-first techniques the frames need to be stored in memory. However, for the frame-first technique the velocity filter states for each pixel need to be stored. This generally requires more memory than storing the frames, thus the pixel-first and filter-first techniques tend be used more in practice.

Several types of simulations were performed to test the performance of the velocity filter with and without noise variance estimation. These included single and multiple targets as well as the production of theoretical ROC curves obtained by executing the velocity filter multiple times. It was found that when the noise variance was estimated the slope of the ROC curves was reduced.

5.1 Further Work

There are many opportunities for further work to be performed in this area. Some of these opportunities are discussed below:

- Simulations were performed for a 1D velocity filter. In practice it is likely that images will be two dimensional thus, the 2D velocity filter would need to be implemented. This would add an extra level of complexity to the simulations and issues such as memory and computation time, which were not crucial in the 1D case, would need to be explored.

- The velocity filter technique could be applied to real data and its performance compared to conventional techniques using this data.
- In this report the issue of estimating the noise variance in order to determine a threshold for detections was investigated. Further work could be performed in this area to find other techniques to estimate the noise variance.
- The concept of data association was introduced in Section 4.1.2, where five detections were reported for only two targets. If the velocity filter technique were to be applied to real data, the way in which detections are associated to individual targets would need to be addressed.
- The velocity filter is one of several track before detect (TBD) techniques (Appendix A). Other TBD techniques could be simulated and their results compared to the velocity filter technique.

6. ACKNOWLEDGEMENTS

I would like to thank my academic supervisor Dr Mehmet Karan. Mehmet's guidance throughout the project was invaluable. I have learnt research skills from Mehmet that will help me in future studies and with my career.

I would also like to thank Dr Anthony Zyweck. Tony's assistance in the early stages of the project is appreciated. I also appreciate the support I received from Tony to assist me in balancing work and study commitments.

Finally I would like to thank the staff of CSSIP and in particular Doug Gray. I found the Masters degree to be valuable and I appreciate the support I received from the CSSIP staff.

7. References

1. S. Kay, 'Fundamentals of Statistical Signal Processing: Detection Theory', Prentice Hall, 1998.

2. I.S. Reed, R.M.Gagliardi and H.M Shao, 'Application of Three-Dimensional Filtering to Moving Target Detection', IEEE Transactions on Aerospace and Electronic Systems Vol. AES 19, No.6, November 1983.

3. A.D. Stocker and P. Jensen, 'Algorithms and Architectures for Implementing Large Velocity Filter Banks', SPIE Vol. 1481 Signal and Data Processing of Small Targets, 1991.

4. P.F. Singer, 'Performance of a Velocity Filter Bank', Proceedings of SPIE - the International Society for Optical Engineering, SPIE Vol. 3163, 1997, USA.

5. S. Blackman and R. Popoli, 'Design and Analysis of Modern Tracking Systems', Artech House, 1999.

6. W.B. Kendall and W.J. Jacobi, 'The Detection and Tracking of Multiple Targets With Velocity Filters', SPIE Vol. 1058 High Speed Computing II, 1989.

7. P.L. Chu, 'Optimum Projection for Multidimensional Signal Detection', IEEE Trans. Acoustics, Speech and Signal Processing, Vol. 36, No. 5, May 1988.

8. S.D. Blostein and T.S. Huang, 'Detection of Small, Moving Objects in Image Sequences Using Multistage Hypothesis Testing', Proceedings ICASSP 1988, Vol. 2, April 1988.

9. E. Kreyszig, 'Advanced Engineering Mathematics, Seventh Edition', John Wiley & Sons, Inc., 1993.

10. B.D. Carlson, 'Search Radar Detection and Track With the Hough Transform', IEEE Trans. on Aerospace and Electronic Systems, Vol. AES-30, Jan. 1994.

11. G.W. Stimson, "Introduction to Airborne Radar", Hughes Aircraft Co., El Segundo, CA, 1983.

Appendix A: Other Track Before Detect Techniques

Velocity filtering is one of many track before detect techniques. In this section two other TBD techniques are discussed, namely the Hough transform and Dynamic Programming.

A.1. Hough Transform

For the Hough transform, measured data is transformed into stationary bins in target state space. In general the bins represent the two normal parameters of a straight line. Reference 5 uses an example of a dim radially inbound target with nearly constant radial velocity. The Hough parameter space (ρ,θ) is mapped into the time-range space using the following equation:

$$\rho = R\cos\theta + t\sin\theta \qquad \text{eq A-1}$$

ρ and θ are broken up into bins which spread over the range and angles that are under consideration. Given a measured range at a given time and angle, Equation A-1 is used to calculate ρ and this measurement is placed in the appropriate (ρ,θ) bin.

A double threshold criterion is used in [10]. A single measurement is not passed to its appropriate (ρ,θ) bin unless the signal power exceeds a primary threshold, T_1. This reduces the amount of noise in each bin at the expense of dismissing some dim signals as noise. After several scans a second threshold, T_2 is applied to each (ρ,θ) bin. If the accumulated signal power in a particular bin exceeds T_2, a detection is declared. For appropriate choices of T_1 and T_2 a double detection threshold increases the probability of detection over a single threshold.

A limitation of the Hough transform is that just like the velocity filter, it assumes that the target moves in a straight line. Targets deviating from this assumption will not be detected as readily. However, Hough transform methods are less computationally intensive than other techniques such as dynamic programming techniques.

A.2. Dynamic Programming Algorithm

As with the Hough transform, the Dynamic Programming Algorithm (DPA) partitions the target state space into bins. For IR applications the typical target states are azimuth and elevation angles and their corresponding rates. The DPA implements the Viterbi algorithm [5] and associates a score for each bin. The score is calculated using the following equation:

Score(j,k) = $\sum$ *{Log-likelihood of Amplitude(j,k) + Log-transition probability(i,j) + Log-prior probability of state(i,k-1)}* *eq A-2*

Where, i and j are the bin numbers and k is the index for a particular scan.

The important difference between the DPA and other TBD methods is the inclusion of the second and third terms in Equation A-2, which enable the DPA to detect manoeuvring targets. Transitions from bin to bin are weighted according to the probability of the target performing the required acceleration. The score function is used to determine the most likely transition between scans for each bin. This transition is recorded for each bin. After N scans the score in each bin is compared to a threshold and should the score exceed the threshold, a target is declared whose trajectory is determined by the stored transitions between the bins. The Viterbi algorithm assumes a first order Markov target motion. Namely, the target motion from scan k-1 to scan k is independent of prior scans.

The disadvantage with the DPA is its complexity and that it is very computationally intensive. This limits the use of the DPA in real time or near real time environments. The obvious advantage of the DPA is its ability to detect manoeuvring targets. With the advent of faster and cheaper processors DPA techniques are becoming more realizable and common.

Appendix B: MATLAB Code

Below is the code from two MATLAB programs. The first 'VF.m', runs the velocity filter bank over the input image a single time. The second 'run_VF.m', executes 'VF.m' in a number of ways. Single or multiple runs can be performed, either one or two targets can be simulated and the user can choose whether to use the real or estimated noise variance.

VF.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Function name: vf
% Programmer: Matthew Dragovic
% Date: 28/10/01
%
% Usage detection = vf(Pfa,Nf,SNR,,no_signals,noise_var_est,ROC)
% Pfa: probability of false alarm in a single velocity filter at a single
%     initial pixel position
% Nf: number of frames
% SNR: Signal to Noise Ratio
% no_signals: number of signals (1 or 2)
% noise_var_est: (0 or 1) 0: use the exact value of the noise variance
%                        1: use an estimated value of the noise varaince
% ROC: (0 or 1) 0: perform a single run analysis
%              1: perform ROC analysis (ie. do not create the plots etc.)
%
% An assumption is made that the target will not leave the field of view
%
% eg detection = vf(10e-4,5,2,1,1)
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function detection = vf(Pfa,Nf,SNR,no_signals,noise_var_est,ROC)

% Noise Statistics
sigma_n_sq = 1;  % noise power
no_pixels = 50;  % number of pixels
noise = randn(Nf,no_pixels)*sqrt(sigma_n_sq);

% Signals
init_loc1 = 10; % initial pixel location of the target (signal)
```

```
vel1 = 0.5;    % velocity of the target (pixels per frame)
sig_pow1 = power(10,SNR/10)*sigma_n_sq;  % signal power

signal1 = zeros(Nf,no_pixels); % initialise the signal matrix
for (j = 1:Nf)
  signal1(j,init_loc1 + round(vel1*(j-1))) = sig_pow1;
end

if (no_signals ~= 1)
  if (no_signals ~= 2)
    disp('Number of signals must be either 1 or 2');
    break;
  end
  init_loc2 = 35; % initial pixel location of the 2nd target
  vel2 = -0.8;    % velocity of the 2nd target
  sig_pow2 = power(10,SNR/10)*sigma_n_sq; % power of the 2nd signal

  signal2 = zeros(Nf,no_pixels); % initialise the 2nd signal matrix
  for (j = 1:Nf)
    signal2(j,init_loc2 + round(vel2*(j-1))) = sig_pow2;
  end
end

% Input to the VF
if (no_signals == 1)
  signals = signal1;
else
  signals = signal1 + signal2;
end

filter_in = noise + signals;

% Implementation of the filter first algorithm
% Assumptions:
% target only occupies 1 pixel

vmin = -1; % minimum velocity of targets the velocity filter bank will cover
vmax = 1;  % maximum velocity of targets the velocity filter bank will cover
delta_v = 1/(Nf-1); % separation between each velocity filter
vel_vector = [vmin:delta_v:vmax];   % the velocity filter bank

pixel_vector = [1:no_pixels];

% initialise power out of each element of the VF bank
h_out = zeros(length(vel_vector),length(pixel_vector));

H = zeros(Nf,no_pixels); % initialise the velocity filter
% The VF is specifically set for each velocity and pixel

for j=[1:length(vel_vector)] % all velocities
  for k=pixel_vector % all pixels
    for n = 1:Nf % all frames
      index = k+round(vel_vector(j)*(n-1)); % index is the pixel position of the velocity filter
                          % in each frame
      if ((index > 0) & (index <= no_pixels))   % within the pixel position bounds
        H(n,index)=1;
      end
    end
    h_out(j,k) = sum(sum(H.*filter_in));   % output of a particular VF (velocity, pixel position)
    H = zeros(Nf,no_pixels);  % reset the VF matrix
  end
end

if ~ROC
  figure(1)
  subplot(211)
  image(100*signals);
  xlabel('Pixel Position')
  ylabel('Frame Number')
  title('Input Signals to the Velocity Filter')

  subplot(212)
  image(filter_in);
  colorbar;
  xlabel('Pixel Position')
  ylabel('Frame Number')
  title('Input Signals + Noise to the Velocity Filter')
end

if (noise_var_est)
  % Estimate the variance of the input noise to the VF

  in_mean_power = 0;
  in_est_var = 0;

  for (j=1:length(filter_in(:,1)))   % all frames
    for (k=1:length(filter_in(1,:)))   % all pixels
      n = k+(j-1)*length(filter_in(1,:));
      if (n > 1)   % n must be greater then zero for the statement below to
                   % execute w/o a divide by zero error. See report for
                   % an explanation of the equation. Basically it is a
                   % recursive estimate of the noise variance.
        if (sqrt(filter_in(j,k)) <= 2*sqrt(in_est_var))
          in_est_var = 1/(n-1)*((n-2)*in_est_var + filter_in(j,k)^2);
        end
      end
    end
  end
  noise_variance = in_est_var;
else
  noise_variance = sigma_n_sq;
end

% Find the threshold for this Pfa and noise variance
threshold = Qinv(Pfa/Nf); % threshold given in number of variances

% find the elements of the VF output matrix that pass the target threshold
% the output is scaled first (see Kay 4.12 to 4.14 for a description of the maths)
% The Nf term must be present as the SNR increases by the sqrt of this term

[a,b]=find(h_out/(noise_variance*sqrt(Nf*power(10,SNR/10)))>threshold);

if ROC
  detection = 0;  % detection is set to 1 if the threshold is passed for the
        % element which corresponds to the signal
  for (j = 1:length(a))
    if (pixel_vector(b(j))==init_loc1)
      if (vel_vector(a(j))-vel1<=delta_v)
        detection = 1;
      end
    end
  end
end

if (~ROC)
  detection_matrix = zeros(length(vel_vector),length(pixel_vector));
  for (j = 1:length(a))
    detection_matrix(a(j),b(j)) = 100;  % set the elements that pass the
        % threshold to a value which allows them to be clearly seen
        % when using the image command
  end

  figure(2)
  subplot(311)
  image(pixel_vector,vel_vector,h_out)
  colorbar;
  xlabel('Pixel Position');
  ylabel('Velocity (Pixels/Frame)')
  title('Filter Output, Velocity-Pixel Matrix')

  subplot(312)
  image(pixel_vector,vel_vector,detection_matrix);
  xlabel('Pixel Position');
  ylabel('Velocity (Pixels/Frame)')
  title('Detection Matrix, Velocity-Pixel Matrix')

  % reconstruct initial image without the noise
  signal = zeros(Nf,no_pixels);
  for (j = 1:length(a))
    est_vel = vel_vector(a(j));
    est_init_loc = b(j);
    for (k = 1:Nf)
      signal(k,est_init_loc + round(est_vel*(k-1))) = 100;
    end
  end

  subplot(313)
  image(signal);
  xlabel('Pixel Position')
  ylabel('Frame Number')
  title('Estimated Signals')
end
```

run_VF.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Function name: run_VF
% Programmer: Matthew Dragovic
% Date: 28/10/01
%
% Usage run_VF(simulation_type)
% simulation_type determines the way in which the vf program will be executed
%
% run_VF(0): single run with 1 signal and no noise variance estimation
% run_VF(1): single run with 2 signals and no noise variance estimation
% run_VF(2): produces a ROC curve for various numbers of frames with no noise variance estimation
% run_VF(3): produces a ROC curve for various numbers of frames with noise variance estimation
% run_VF(4): produces a ROC curve for various SNR with no noise variance estimation
% run_VF(5): produces a ROC curve for various SNR with noise variance estimation
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function run_Vf(simulation_type)

  switch simulation_type
  case 0
    vf(10^-4,10,3,1,0,0);
  % single run with 1 signal and no noise variance estimation
  case 1
    vf(10^-4,10,5,2,0,0);
  % single run with 2 signals and no noise variance estimation
  case 2
    VF_ROC(500,0,0.1,0);
  % produces a ROC curve for various numbers of frames with no noise variance estimation
  case 3
    VF_ROC(500,0,0.1,1);
  % produces a ROC curve for various numbers of frames with noise variance estimation
  case 4
    VF_ROC(500,1,0.1,0);
  % produces a ROC curve for various SNR with no noise variance estimation
  case 5
    VF_ROC(500,1,0.1,1);
  % produces a ROC curve for various SNR with noise variance estimation
  otherwise
    disp('usage: run_VF(value) where value is 0,1,2,3,4 or 5')
  end
end

function VF_ROC(N,Nf_or_SNR,delta_Pfa,noise_var_est)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Usage VF_ROC(N,Nf_or_SNR,delta_Pfa,noise_var_est)
%
% N:            number of times VF.m is executed for each point on each curve
% Nf_or_SNR:    0 if we want curves for different numbers of frames
%               1 if we want curves for various SNR
% delta_Pfa:    space between each Pfa value (log10 base)
% noise_var_est:        0 use the real value of the noise variance
%               1 estimate the noise variance
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  Pfa_vector = fliplr([1:delta_Pfa:6]);
  Nf_vector = [2 3 5 7 10];
  SNR_vector = [0 1 2 3 4 5];
```

```
 if (Nf_or_SNR)
   detection_count = zeros(length(Pfa_vector),length(SNR_vector));
 else
   detection_count = zeros(length(Pfa_vector),length(Nf_vector));
 end

 for (j = 1:length(Pfa_vector))
   for (k = 1:N)
     if (Nf_or_SNR)
       for (m = 1:length(SNR_vector)) % various SNR
         detection_count(j,m)        =        detection_count(j,m)        +        vf(10^-
Pfa_vector(j),5,SNR_vector(m),1,noise_var_est,1);
       end
     else
       for (m = 1:length(Nf_vector)) % various Nf
         detection_count(j,m)        =        detection_count(j,m)        +        vf(10^-
Pfa_vector(j),Nf_vector(m),10*log10(4),1,noise_var_est,1);
       end
     end
   end
 end

 Pd = detection_count/N; % probability of detection

 xaxis = [10.^-Pfa_vector];
 semilogx(xaxis,Pd);
 grid on
 xlabel('Probability of False Alarm')
 ylabel('Probability of Detection')

 % insert the legend information
 string_matrix = '';
```

```
  if (Nf_or_SNR)
    title('Velocity Filter Bank ROC Curve for Various SNR')
    for (j = 1:length(SNR_vector))
      string_matrix = [string_matrix;sprintf('SNR = %d',SNR_vector(j))];
    end
  else
    title('Velocity Filter Bank ROC Curve for Various Numbers of Frames')
    for (j = 1:length(Nf_vector))
      if (Nf_vector(j) < 10)
        string_matrix = [string_matrix;sprintf('Nf = %d ',Nf_vector(j))];
      else
        string_matrix = [string_matrix;sprintf('Nf = %d',Nf_vector(j))];
      end
    end
  end

  legend(string_matrix,0);

end
```

DISTRIBUTION LIST

Velocity Filtering for Target Detection and Track Initiation

Matthew Dragovic

AUSTRALIA

DEFENCE ORGANISATION

Task Sponsor
DDCE, WGCDR Phil Lavelle

S&T Program
Chief Defence Scientist
FAS Science Policy } shared copy
AS Science Corporate Management
Director General Science Policy Development
Counsellor Defence Science, London (Doc Data Sheet)
Counsellor Defence Science, Washington (Doc Data Sheet)
Scientific Adviser Joint
Navy Scientific Adviser (Doc Data Sheet and distribution list only)
Scientific Adviser - Army (Doc Data Sheet and distribution list only)
Air Force Scientific Adviser
Scientific Adviser to the DMO (Doc Data Sheet and distribution list only)
Director Trials
A7 STCC Dr Douglas Kewley (Doc Data Sheet)

Systems Sciences Laboratory

Weapon Systems Division
CWSD, Mr Bill Dickson (Doc Data Sheet)
RLAWS, Dr Douglas Kewley
RLMWS, Dr Colin Coleman
HRFS, Dr Anthony Szabo
Task Manager, Dr Anthony Zyweck
Author, Mr Matthew Dragovic

Air Operations Division
RLAOA, Mr James Smith
HOPO, Dr Torgny Josefsson

Electronic Warfare & Radar Division
CEWRD, Dr Len Sciacca
RLMR, Dr Andrew Shaw
RLEW, Dr Tony Lindsay
RLRFEW, Dr Peter Gerhardy
HAR, Dr John Whitrow
HRFCM, Mr Mike Bell
Dr Bob Warren
Mr John Baldwinson

DSTO Library and Archives

Library Edinburgh (2 copies)
Australian Archives

Capability Systems Division

Director General Maritime Development (Doc Data Sheet)
Director General Information Capability Development (Doc Data Sheet)
Director General Aerospace Development (Doc Data Sheet)

Office of the Chief Information Officer

Chief Information Officer (Doc Data Sheet)
Deputy CIO (Doc Data Sheet)
Director General Information Policy and Plans (Doc Data Sheet)
AS Information Structures and Futures (Doc Data Sheet)
AS Information Architecture and Management (Doc Data Sheet)
Director General Australian Defence Information Office (Doc Data Sheet)
Director General Australian Defence Simulation Office (Doc Data Sheet)

Strategy Group

Director General Military Strategy (Doc Data Sheet)
Director General Preparedness (Doc Data Sheet)

HQAST

SO (ASJIC)

Army

ABCA National Standardisation Officer, Land Warfare Development Sector, Puckapunyal (4 copies)
SO (Science), Deployable Joint Force Headquarters (DJFHQ) (L), Enoggera QLD (Doc Data Sheet)
SO (Science) - Land Headquarters (LHQ), Victoria Barracks NSW (Doc Data Sheet and Executive Summary Only)

Intelligence Program

DGSTA Defence Intelligence Organisation
Manager, Information Centre, Defence Intelligence Organisation

Defence Libraries

Library Manager, DLS-Canberra
Library Manager, DLS - Sydney West (Doc Data Sheet)

Defence Materiel Organisation

Project Manager AIR 5400
Head Airborne Surveillance and Control (Doc Data Sheet)
Head Aerospace Systems Division (Doc Data Sheet)
Head Electronic Systems Division (Doc Data Sheet)
Head Maritime Systems Division (Doc Data Sheet)
Head Land Systems Division (Doc Data Sheet)

UNIVERSITIES AND COLLEGES

Australian Defence Force Academy
- Library
- Head of Aerospace and Mechanical Engineering

Serials Section (M list), Deakin University Library, Geelong, VIC
Hargrave Library, Monash University (Doc Data Sheet)
Librarian, Flinders University
Deputy CEO CSSIP, Prof. Doug Gray

OTHER ORGANISATIONS

National Library of Australia
NASA (Canberra) (Doc Data Sheet)
State Library of South Australia

OUTSIDE AUSTRALIA

INTERNATIONAL DEFENCE INFORMATION CENTRES

US Defense Technical Information Center,	(2 copies)
UK Defence Research Information Centre, (2 copies)	(2 copies)
Canada Defence Scientific Information Service	(1 copy)
NZ Defence Information Centre	(1 copy)

INFORMATION EXCHANGE AGREEMENT PARTNERS

Acquisitions Unit, Science Reference and Information Service, UK

SPARES (5 copies)

Total number of copies: 51

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION **DOCUMENT CONTROL DATA**	1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)
2. TITLE Velocity Filtering for Target Detection and Track Initiation	3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION) Document (U) Title (U) Abstract (U)
4. AUTHOR(S) Matthew Dragovic	5. CORPORATE AUTHOR Systems Sciences Laboratory PO Box 1500 Edinburgh South Australia 5111 Australia

6a. DSTO NUMBER DSTO-TR-1406	6b. AR NUMBER AR-012-603	6c. TYPE OF REPORT Technical Report	7. DOCUMENT DATE March 2003

8. FILE NUMBER J 9505-23-44 Pt 1	9. TASK NUMBER AIR 01/006	10. TASK SPONSOR DGAD	11. NO. OF PAGES 59	12. NO. OF REFERENCES 11

13. URL on the World Wide Web http://www.dsto.defence.gov.au/corporate/reports/DSTO-TR-1406.pdf	14. RELEASE AUTHORITY Chief, Weapons Systems Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for public release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS Yes

18. DEFTEST DESCRIPTORS

Tracking Filters, Target Acquisition, Signal to Noise Ratio

19. ABSTRACT

The velocity filter is a variation of the 3D matched filter. Velocity filtering applies a constraint in the form of assuming that targets will have a constant velocity over the integration period of the filter. Velocity filters are applied over multiple frames of data and are able to detect low Signal-to-Noise Ratio (SNR) targets that would otherwise be undetectable using conventional 'single look' detection techniques. This report derives, discusses and assesses the performance of the velocity filter technique.